NSWC/WOL/TR 77-28

NSWC/WOL/TR 77-28

NSWC TECHNICAL REPORT

WHITE OAK LABORATORY

A PROGRAM FOR COMPUTING STEADY INVISCID THREE-DIMENSIONAL SUPERSONIC FLOW ON REENTRY VEHICLES, VOL I: ANALYSIS AND PROGRAMMING

11 FEBRUARY 1977

NAVAL SURFACE WEAPONS CENTER
WHITE OAK LABORATORY
SILVER SPRING, MARYLAND 20910

- Approved for public release; distribution unlimited.

NAVAL SURFACE WEAPONS CENTER
WHITE OAK, SILVER SPRING, MARYLAND 20910

UNCLASSIFIED

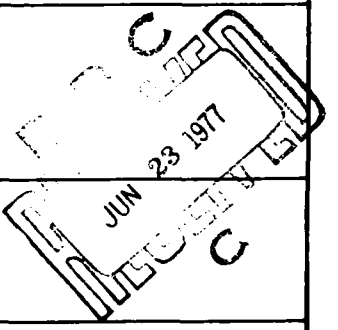
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: NSWC/WOL/TR-77-28	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): A Program for Computing Steady Inviscid Three-Dimensional Supersonic Flow on Reentry Vehicles, Volume I: Analysis and Programming.	5. TYPE OF REPORT & PERIOD COVERED / 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): J. M. Solomon, M. Ciment, R. E. Ferguson, J. B. Bell, A. B. Wardlaw, Jr.	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Surface Weapons Center, White Oak Laboratory, White Oak, Silver Spring, Maryland 20910	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 0; 0; 0; 0;
11. CONTROLLING OFFICE NAME AND ADDRESS	12. REPORT DATE: 11 February 1977 / 13. NUMBER OF PAGES: 207
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified / 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)



18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

reentry vehicle, flow fields, three dimensional, supersonic/hypersonic, inviscid flow, control surfaces, pitch/yaw, finite difference methods, hyperbolic systems, boundary conditions

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A comprehensive computational procedure is presented for predicting the supersonic region of the flow field on advanced reentry vehicle shapes in steady flight at pitch and yaw. The procedure utilizes explicit second order accurate finite difference methods applied to the conservation law form of the steady inviscid flow equations. Improved numerical methods are used at the body surface and the bow shock wave. Provisions for treating body

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

geometries with discontinuous slopes are also included. Either perfect gas or real gas equilibrium thermodynamic properties can be used.

The computational procedure is implemented as a fortran computer code which provides a practicable representation of the inviscid flow field and the resulting aerodynamic force and moment on the vehicle.

In this report ~~(Vol. I)~~ the analytical and numerical development of the procedure is presented and the associated computer code is described. A comparison report ~~(Vol. II User's Manual)~~ contains detailed instructions for operating the code and interpreting the output results.

NSWC/WOL/TR 77-28 11 February 1977

A Program for Computing Steady Inviscid Three-Dimensional Supersonic Flow on Reentry Vehicles, Vol. I: Analysis and Programming

This report describes the analytical, and computational aspects of a computer program for predicting inviscid flow fields and aerodynamics on realistic reentry configurations. This work was performed by members of the Mathematical and Engineering Analysis Branch of NSWC/WOL. The initial code development was supported by the Naval Sea Systems Command under the Aeroballistic Reentry Technology (ART) program with some of the fundamental analytical and numerical work supported by NSWC Independent Research Funds. Most of the final code development and documentation was supported by the Air Force Space and Missile System Organization under the technical management of the Aerospace Corporation.

The authors gratefully acknowledge the efforts of Mr. R. Feldhuhn, NSWC coordinator for the ART program, who was responsible for initiating the present work and whose continued interest and support throughout the investigation was invaluable. The authors are also indebted to Mr. M. Lyons and Dr. E. Ndefo of the Aerospace Corporation for several stimulating technical discussions which lead to important improvements in the final code.

C. A. Fisher

C. A. FISHER
By direction

CONTENTS

Page

0. INTRODUCTION

An important aspect of the design and evaluation of maneuverable and advanced ballistic reentry vehicles is the determination of the inviscid flow field surrounding the body. The inviscid flow field provides surface pressure distributions required for determining the aerodynamic loading on the vehicle and other surface information which is needed as input for determining surface heat transfer rates and other boundary layer effects. A cost effective method for obtaining this information is to use high-speed computer codes which numerically solve the steady, three-dimensional, inviscid flow equations associated with arbitrary shaped reentry vehicles flying at supersonic/hypersonic speeds.

The numerical calculation of the inviscid flow field over reentry vehicles is divided into two parts--the blunt body region calculation and the supersonic region calculation (see Fig. 1). The blunt body region calculation determines the transonic flow field near the stagnation point. The supersonic region calculation determines the flow field downstream of the blunt body region. The differing nature of the flow in these two regions requires significantly different computer codes for calculating each region. The blunt body region is computed first and is continued downstream until supersonic flow is established everywhere in the shock layer. The computed results from this calculation are used to establish an "initial" data plane which is used to start the supersonic region calculation. This latter portion represents the majority of the total flow field on maneuverable and high performance ballistic reentry vehicles.

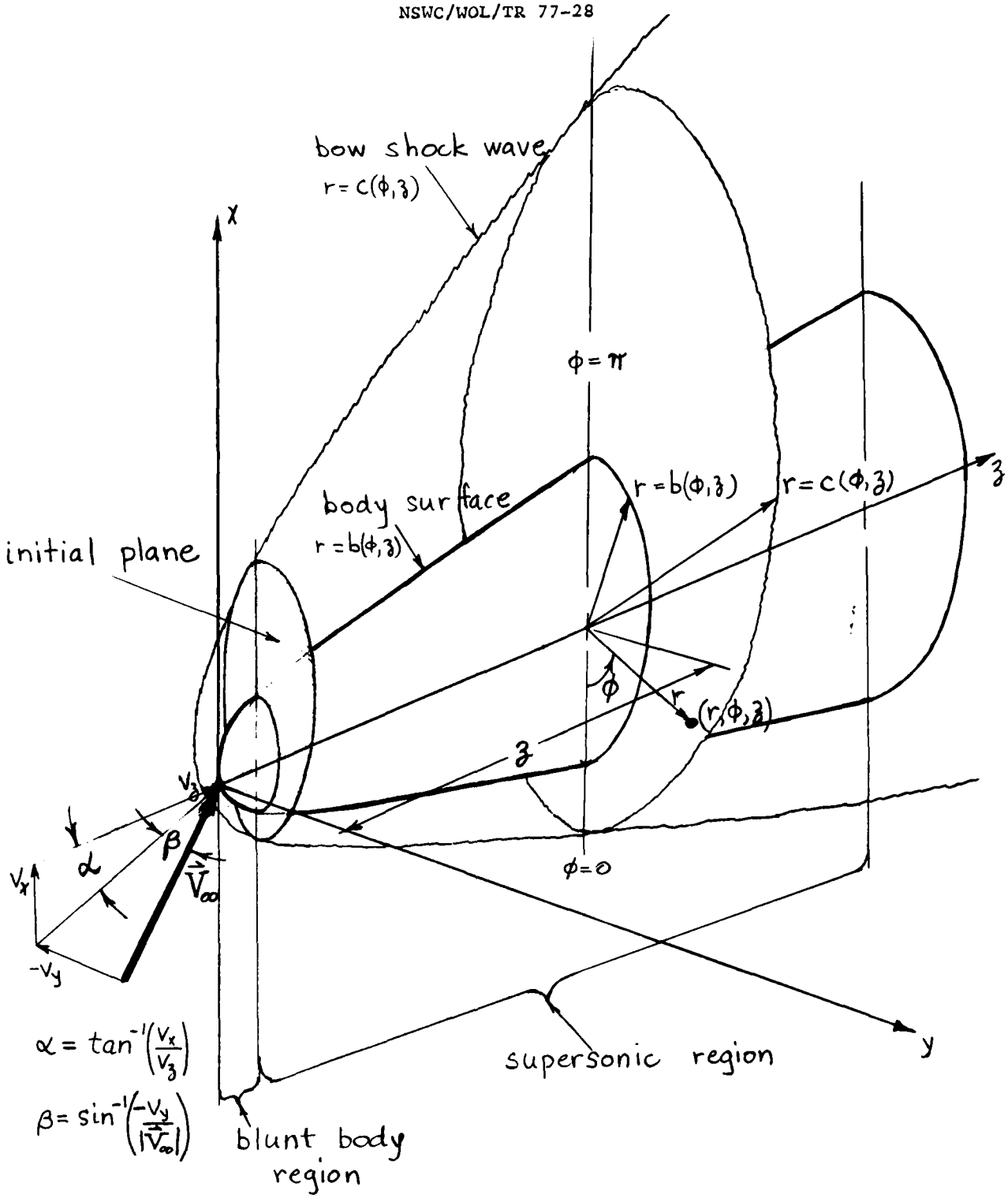


Fig. 1. Computational regions and cylindrical coordinate system for reentry vehicle inviscid flow calculations.

In this report, a computer code for performing the supersonic region calculation is described. This code is based on the conservation law form of the steady, inviscid equations. Codes of this type are sometimes referred to as shock capturing techniques since internal (embedded) shock waves in the flow field are computed in an approximate manner without explicitly locating (or tracking) the shock surface. The present code incorporates improved numerical methods at the body surface and the bow shock wave which yield a wider applicability to missile design than existing codes of this type (e.g. ref. 1).

This report is divided into two parts. In Part I, the partial differential equations, boundary conditions, and finite difference equations which are the basis of the computer code are discussed. In Part II, the fortran computer code is described. A companion report*, the Users' Manual, contains detailed instructions for running the code on CDC 6000 series and 7600 machines and for interpreting the output results.

*Solomon, J. M., Ciment, M., Ferguson, R. E., Bell, J. B., and Wardlaw, A. B., A Program for Computing Steady Inviscid Three-Dimensional Supersonic Flow on Reentry Vehicles - Vol. II: User's Manual, NSWC/WOL TR 77-32

PART I: ANALYSIS

1. NOTATION (PART I)

Symbols

a	sound speed
a_3, a_4, a_5, a_7	body geometry parameters; see, (3.16a) and (3.16b)
A	see (3.3d)
A_1, A_0	see (3.9c)
A_{ref}	reference area for force and moment coefficients nondimensionalized by R_o^2 (see Fig. 7)
$\mathbb{A}$	see (A-5a)
A^*	characteristic matrix (Appendix A)
$b = b(\phi, z)$	body shape function
B	see (3.3d)
$\mathbb{B}$	see (A-5b)
$c = c(\phi, z)$	shock shape function
C	see Fig. 7
C_1, C_2	see (3.9c)
$\mathfrak{C}$	step size factor, see (3.25)
D	see (3.22)
$\vec{e}_z, \vec{e}_r, \vec{e}_\phi$	unit vectors in z,r,ϕ directions, respectively
$\vec{e}_\sigma, \vec{e}_\tau, \vec{e}_n$	see Fig. 4 and (4.3)
e_1	see (3.3e)
E	see (3.3c)
$\mathfrak{E}$	see (2.1b)
$f = f(X,Y,Z)$	clustering transformation, see (3.2)
F	see (3.3a)
F_a, F_n, F_y	aerodynamic force components, see Fig. 7 and (5.1) - (5.3)
$\mathfrak{F}$	see (2.1b)

$g = g(Y,Z)$	clustering transformation, see (3.2)
G	see (3.3b)
$\mathcal{G}$	see (2.1b)
h	enthalpy,
H_∞	total enthalpy, see (2.2)
H_o	see (3.8b)
j	0 or 1, determines scheme, see (3.6a) and (3.6b)
k	step number
$\mathcal{K}_1, \mathcal{K}_2$	thermodynamic derivatives, see (3.16) and (A-4)
$\ell_\lambda, \ell_{\lambda_+}, \ell_{\lambda_o}^{(1)}, \ell_{\lambda_o}^{(2)}$	characteristic null vectors, see Appendix A
m	denotes mesh point $Y = Y_m$, see (3.4)
ms	limit mesh point for wall entropy reduction, see Sec. 4.2
M	value of m corresponding to Y = 1, see Figs. 2 and 3
M_∞	free stream Mach number
M_a^c, M_n^c, M_y^c	aerodynamic moment components about C, see Fig. 7 and (5.4) - (5.6)
$\mathcal{M}$	Mach number in front of oblique shock, see (4.9)
n	denotes mesh point $X = X_n$, see (3.4)
$\vec{n}$	normal vector
N	value of n corresponding to X = 1 (bow shock), see Figs. 2 and 3
$\mathcal{O}$	matrix, see Appendix A
p	pressure
P	log p
$\mathcal{P}$	see (3.16a) and (3.16b)

$\tilde{\mathcal{P}}$	see (3.19)
Q	(p,u,v,w)
r	radial coordinate, nondimensionalized by R_o
R_o	nondimensionalizing length
$\mathcal{R}, \tilde{\mathcal{R}}$	see Appendix A
$\mathcal{R}_s$	see (3.9c)
s	entropy,
$T_{g_5}, T_{f_4}, T_{f_6} T_{f_7}$	transformation quantities, see (3.3h) and (3.3i)
u	velocity component in r direction
$U = (U_1, U_2, U_3, U_4)$	conservation vector, see (2.1a)
$\tilde{U}$	averaged values of U, see (4.11)
v	velocity component in ϕ direction
V	$\sqrt{u^2 + v^2 + w^2}$
$\vec{V}$	velocity vector
V_n	normal component of velocity
V_σ	velocity component in σ direction
V_2	$v + (b_\phi/b)u$
$\mathcal{V}$	see (3.17a) and (3.17b)
w	velocity component in z direction
x,y	see Fig. 1 or Fig. 7
$(\bar{x}, \bar{y}, \bar{z})$	see (3.1)

(X,Y,Z)	coordinates in computational space, see Figs. 2 & 3
X_z, X_r, X_ϕ	see (3.3g)
Y_z, Y_ϕ	see (3.3f)
z	axial coordinate (see Figs. 1 or 6) nondimensionalized by R_o
z_c	z coordinate of moment center C, see Fig. 7
z_{ref}	reference length for moment coefficients, nondimensionalized by R_o
z_o	axial location of initial plane
$(z_{c.p.})_p$	center of pressure in pitch, see (5.9)
$(z_{c.p.})_y$	center of pressure in yaw, see (5.10)
α	angle of attack, see Fig. 1
β	angle of side slip, see Fig. 1
β_0	see (3.9c)
β_1	see (3.16)
γ	ratio of specific heats
Γ	effective gamma, see (3.7)
δ	$\Delta X/\Delta Y$ $(j = 0)$, $-\Delta X/\Delta Y$ $(j = 1)$
$\Delta X, \Delta Y$	computational mesh spacing, $\Delta X = 1/(N-1)$, $\Delta Y = 1/(M-1)$
ΔZ	step size, see (3.25)
$\eta = (\eta_1, \eta_2, \eta_3, \eta_4)$	see (3.17a)
θ	turning angle, see (4.7)
Θ	see (4.11)
$\zeta = (\zeta_1, \zeta_2, \zeta_3, \zeta_4)$	see (3.9c)
$\lambda, \lambda_o, \lambda_\pm$	characteristic values, see Appendix A
λ_+	see (3.16)
λ_-	see (3.9c)
μ, μ_1, μ_2, μ_3	stability parameters, see Sec. 3.6

ν_w	$\sqrt{1 + (b_\phi/b)^2 + b_z^2}$
ν_∞	$\sqrt{1 + (c_\phi/c)^2 + c_z^2}$
$\xi = (\xi_1,\xi_2,\xi_3,\xi_4)$	see (3.16a)
ρ	density
$\vec{\sigma},\vec{\tau}$	see Fig. 4
ϕ	azimuthal angle
ϕ_o	π for the symmetric problem; 2π for the nonsymmetric problem
Φ	see (3.8c)

Subscripts

n,m	quantity evaluated at $X = X_n$, $Y = Y_m$
s	quantity evaluated at bow shock wave
w	quantity evaluated at wall
$\overline{+}$	quantities at upstream and downstream side of body slope discontinuity in section 4.1 only

Superscripts

k	quantity at step $Z = Z^k$
*	predicted value (except in Appendix A)

Other

A dependent variable used as a subscript denotes partial differentiation with respect to that variable; e.g., if $b = b(\phi,z)$ then $b_z = \frac{\partial b}{\partial z}$.

$|q|$ denotes the absolute value of q if q is a scalar and the modulus (or magnitude) of q if q is a vector.

$\cdot$ and $\times$ denote vector inner product and cross products, respectively.

2. GOVERNING EQUATIONS AND BOUNDARY CONDITIONS

2.1 Steady Flow Equations

Consider a body oriented cylindrical coordinate system r,ϕ,z illustrated in Figure 1. With respect to these coordinates, the conservation of mass and momentum equations for a steady, inviscid, flow can be written as a system of (weak) conservation laws; i.e.,

$$\frac{\partial U}{\partial z} + \frac{\partial \mathcal{F}}{\partial r} + \frac{\partial \mathcal{G}}{\partial \phi} + \mathcal{E} = 0 \tag{2.1}$$

where

$$U = \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{pmatrix} = \begin{pmatrix} \rho w \\ p + \rho w^2 \\ \rho wu \\ \rho wv \end{pmatrix} \tag{2.1a}$$

and

$$\mathcal{F} = \begin{pmatrix} \rho u \\ \rho wu \\ p + \rho u^2 \\ \rho vu \end{pmatrix}, \quad \mathcal{G} = \frac{1}{r}\begin{pmatrix} \rho v \\ \rho vw \\ \rho vu \\ p + \rho v^2 \end{pmatrix}, \quad \mathcal{E} = \frac{1}{r}\begin{pmatrix} \rho u \\ \rho uw \\ \rho(u^2-v^2) \\ 2\rho uv \end{pmatrix}. \tag{2.1b}$$

In the above, p denotes pressure and ρ denotes density and u,v,w are the velocity components in the r,ϕ,z directions, respectively. The energy equation for a steady inviscid flow with an isoenergetic free stream can be reduced to an algebraic equation; viz.

$$h + \frac{1}{2}(u^2+v^2+w^2) = H_\infty \equiv h_\infty + \frac{1}{2}(u_\infty^2+v_\infty^2+w_\infty^2). \tag{2.2}$$

Here h is the enthalpy and H_∞ is the total energy in the free stream (which is constant). In the above equations and throughout our discussion, all lengths are non-dimensionalized by R_o, some body length; all other quantities are dimensional.*

*It is understood that the dimensions are consistent, i.e., velocity has dimensions of (pressure/density)$^{\frac{1}{2}}$ and enthalpy has dimensions of pressure/density.

The flow is assumed to be in local thermodynamic equilibrium (or frozen-equilibrium) so that known relationships exist between the thermodynamic variables, p, ρ, and h. Two other thermodynamic variables will be introduced later; viz., the sound speed, a, and the entropy, s. We assume the standard form of the thermodynamic relations which expresses h, ρ, and a as functions of p and s. In addition, we assume that s can be expressed as a function of p and h and also as a function of p and ρ. The computational algorithm which will be described in the next section specifically requires that $h = h(p,\rho)$, $a = a(p,\rho)$ and, at certain points, $\rho = \rho(p,s)$--these relations being given either in closed form or as curve fits. Note that the sound speed can be defined in terms of $h(p,\rho)$ by

$$a^2 \left(\frac{\partial h}{\partial p}\right)_\rho + \left(\frac{\partial h}{\partial \rho}\right)_p = \frac{a^2}{\rho} . \tag{2.3}$$

For the special case of a perfect gas* with ratio of specific heats, γ, we have

$$h = \frac{\gamma}{\gamma-1}\frac{p}{\rho} , \quad a^2 = \gamma \frac{p}{\rho} \qquad [\text{perf}] \tag{2.4}$$

and

$$\rho/\rho_\infty = \exp\left\{\frac{1}{\gamma}\left(\log\left(p/p_\infty\right) - (s - s_\infty)/c_v\right)\right\}, \quad [\text{perf}] \tag{2.5}$$

where c_v is the specific heat at constant volume.

*The term perfect gas used herein refers to a gas satisfying the perfect gas law (thermodynamically perfect) and having constant specific heats (calorically perfect). Throughout this report equations which are valid only for perfect gases are followed by [perf].

2.2 Boundary Conditions

The shock layer is bounded by the given body surface, and the bow shock wave, which is an unknown to be determined. The body surface is assumed to be prescribed in the form $r = b(\phi,z)$ where the function $b(\phi,z)$ is continuous and piecewise twice continuously differentiable. On the body surface, the boundary condition for inviscid flow is that the normal component of velocity must vanish, i.e.,

$$u - b_z w - (b_\phi/b)v = 0 \tag{2.6}$$

The bow shock surface is assumed to be in the form $r = c(\phi,z)$, an unknown function, to be determined in the calculation. On the bow shock surface the Rankine-Hugoniot relations must hold. These relations are:

$$\begin{aligned} \rho V_n &= \rho_\infty V_{n_\infty} , \\ \vec{V} - \vec{n}V_n &= \vec{V}_\infty - \vec{n}V_{n_\infty} , \\ p + \rho V_n^2 &= p_\infty + \rho_\infty V_{n_\infty}^2 , \\ h + \tfrac{1}{2}V_n^2 &= h_\infty + \tfrac{1}{2}V_{n_\infty}^2 . \end{aligned} \tag{2.7}$$

In the above, $V_n = \vec{n}\cdot\vec{V}$ is the component of velocity normal to the shock surface and, $\vec{V}_t = \vec{V} - \vec{n}V_n$ where $\vec{n}$ is the unit vector normal to the shock surface given by

$$\begin{aligned} \vec{n} &= \frac{1}{\nu_s}\left(c_z \vec{e}_z - \vec{e}_r + \frac{c_\phi}{c}\vec{e}_\phi\right) , \\ \nu_s &= \sqrt{1 + c_z^2 + (c_\phi/c)^2} \end{aligned} \tag{2.7a}$$

where $\vec{e}_z$, $\vec{e}_r$, and $\vec{e}_\phi$ are the unit vectors in the z, r, and ϕ directions, respectively. In (2.7), the subscript ∞ refers to the free stream quantities (upstream of the shock); all other quantities are values immediately behind (downstream of) the shock.

3. BASIC COMPUTATIONAL ALGORITHM

The region of the shock layer in which the axial velocity component is supersonic (i.e., w>a) is referred to here as the supersonic region. It can be shown that in this region the system (2.1) is of hyperbolic type with the z axis as the time-like direction. This implies among other things that the numerical solution of (2.1), (2.2) can be obtained by marching stepwise in the z direction. The procedure adopted here employs an explicit finite difference method which advances (i.e., determines) the flow variables ρ,p,w,u,v and the bow shock geometry c, c_r, c_z to station $z + \Delta z$ using the known quantities at station z. The step size Δz is chosen to satisfy a stability criterion.* The calculation begins at some plane $z = z_o$ in the supersonic region where all flow quantities and shock geometry are known. This plane will be referred to as the initial plane. The calculation is continued downstream to any desired axial station (or until the axial velocity becomes sonic) by repeated use of the procedure using the previous $z + \Delta z$ as the next known station.

3.1 Computational Region and Transformed Equations

In the following discussions two different problems will be considered; the symmetric and the non-symmetric problems. In the symmetric problem, the body is assumed to be symmetric about the pitch plane (i.e., in Fig. 1, the body is symmetric about the x - z plane and $\beta = 0$). The calculation for this problem need only be performed for $0 \leq \phi \leq \pi$ since the flow field is symmetric about $\phi = 0$ and $\phi = \pi$ (wind and lee sides, respectively).

*See Section 3.6

The planes $\phi = 0$ and $\phi = \pi$ are symmetry boundaries where all variables are even functions of ϕ except v which is an odd function of ϕ. In the non-symmetric problem, the body need not be symmetric and/or $\beta \neq 0$. In this problem, the calculation must be performed for $0 \leq \phi \leq 2\pi$; and all variables are periodic functions of ϕ with period 2π.

For both problems, the shock layer for $z \geq z_o$ is transformed into the computational region $Z \geq z_o$, $0 \leq X \leq 1$, $0 \leq Y \leq 1$. The transformation is conveniently expressed as a composite of two mappings. The first is given by the usual normalizing transformations;

$$\begin{aligned} \bar{z} &= z \\ \bar{x} &= [r-b(z,\phi)]/[c(z,\phi)-b(z,\phi)] \\ \bar{y} &= \phi/\phi_o \end{aligned} \tag{3.1}$$

where ϕ_o is π for the symmetric problem and 2π for the non-symmetric problem. This mapping by itself transforms the shock layer into the region $\bar{z} \geq z_o$, $0 \leq \bar{x} \leq 1$, $0 \leq \bar{y} \leq 1$. The second mapping maps this region one-to-one onto itself. For computational purposes, it is convenient to express the second mapping in inverted form, i.e.,

$$\begin{aligned} \bar{z} &= Z \\ \bar{x} &= f(X,Y,Z) \\ \bar{y} &= g(Y,Z) \end{aligned} \tag{3.2}$$

where $f(0,Y,Z) = 0$, $f(1,Y,Z) = 1$; $g(0,Z) = 0$, $g(1,Z) = 1$. The primary purpose of the second mapping is to cluster computational points in the shock layer by choosing f and g appropriately (see sec. 4.3 for a discussion of this feature). The mapping functions $f(X,Y,Z)$ and $g(Y,Z)$ must be

twice continuously differentiable and be given so that the functions and their derivatives up to and including the second order are accordingly smoothly defined. Apart from this restriction (and certain ones which will be imposed in sec 3.5), the functions $f(X,Y,Z)$ and $g(Y,Z)$ can be arbitrary. When no transformation is desired, then one should set $f \equiv X$ and $g \equiv Y$.

The governing equations, (2.1), when transformed to the computational space (X,Y,Z) by (3.1) - (3.2) become

$$\frac{\partial U}{\partial Z} + \frac{\partial F}{\partial X} + \frac{\partial G}{\partial Y} + E = 0 \tag{3.3}$$

where

$$F = X_z U + X_r \mathcal{F} + X_\phi \mathcal{G} = \begin{pmatrix} \rho A \\ X_z p + \rho wA \\ X_r p + \rho uA \\ \frac{1}{r} X_\phi p + \rho vA \end{pmatrix} \tag{3.3a}$$

$$G = Y_z U + Y_\phi \mathcal{G} = \begin{pmatrix} \rho B \\ Y_z p + \rho wB \\ \rho uB \\ \frac{1}{r} Y_\phi p + \rho vB \end{pmatrix} \tag{3.3b}$$

$$E = \mathcal{E} - [(\frac{\partial X_z}{\partial X} + \frac{\partial Y_z}{\partial Y})U + \frac{\partial X_r}{\partial X}\mathcal{F} + (\frac{\partial X_\phi}{\partial X} + \frac{\partial Y_\phi}{\partial Y})\mathcal{G}]$$

$$= \begin{pmatrix} e_1 \\ we_1 + (Y_z T_{g_5} + T_{f_4} X_z + T_{f_6})p \\ ue_1 - \frac{\rho v^2}{r} + X_r T_{f_4} p \\ ve_1 + \frac{1}{r}[\rho vu + p(X_\phi T_{f_4} + Y_\phi T_{g_5} + T_{f_7})] \end{pmatrix} . \tag{3.3c}*$$

*The awkward appearing notation (T_{f_4}, T_{f_6} etc.) follows the FORTRAN formulation of our code.

In the above,

$$A = X_z w + X_r u + \frac{1}{r} X_\phi v \ , \quad B = Y_z w + \frac{1}{r} Y_\phi v \tag{3.3d}$$

$$e_1 = \rho\left(\frac{u}{r} + AT_{f_4} + BT_{g_5} + wT_{f_6} + \frac{v}{r} T_{f_7}\right) \tag{3.3e}$$

$$Y_z = - g_Z/g_Y \ , \quad Y_\phi = 1/(\phi_o g_Y) \tag{3.3f}$$

$$\left.\begin{aligned} X_r &= 1/[f_X(c-b)] \\ X_z &= - (f_Z + Y_z f_Y)/f_X + X_r[(f-1)b_z - fc_z] \\ X_\phi &= - Y_\phi f_Y/f_X + X_r[(f-1)b_\phi - fc_\phi] \end{aligned}\right\} \tag{3.3g}$$

$$T_{g_5} = g_{YY}/g_Y \ , \ T_{g_6} = g_{ZY}/g_Y \tag{3.3h}$$

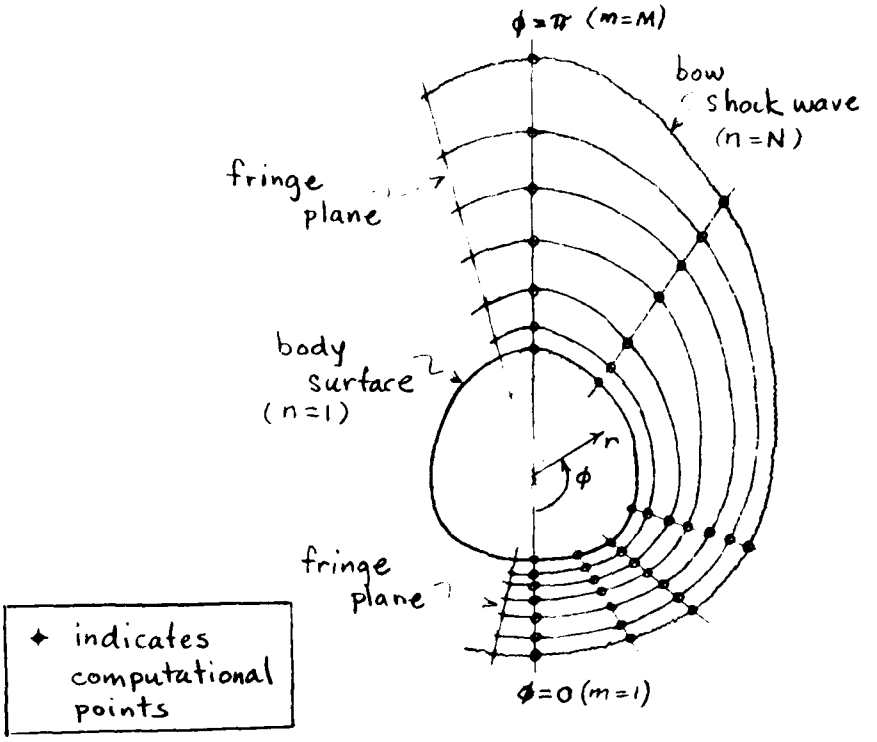
$$\left.\begin{aligned} T_{f_4} &= f_{XX}/f_X \ , \quad T_{f_7} = Y_\phi f_{YX}/f_X - (b_\phi - c_\phi)/(c-b) \\ T_{f_6} &= T_{g_6} + (f_{ZX} + Y_Z f_{YX})/f_X - (b_z - c_z)/(c-b) \end{aligned}\right\} \tag{3.3i}$$

Note that in the above, the body slopes, b_z and b_ϕ, are derived from the given body geometry function; the shock slopes, c_z and c_ϕ, are unknowns to be determined in the calculation.

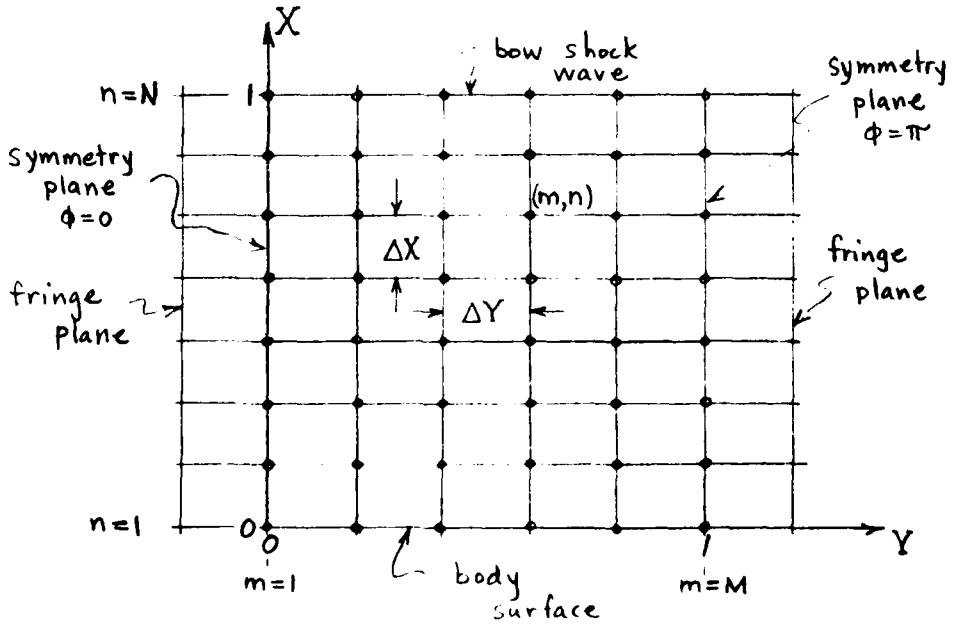
The system of partial differential equations (3.3) is discretized and solved numerically in the computational space using a mesh defined by

$$\{(X_n, Y_m): X_n = (n-1)\Delta X \ (n=1,2,\cdots,N), \ Y_m = (m-1)\Delta Y (m=1,2,\cdots M)\} \tag{3.4}$$

where $\Delta X = 1/(N-1)$ and $\Delta Y = 1/(M-1)$. In Figures 2 and 3, we depict typical discretized computational planes $Z = z =$ constant and the corresponding

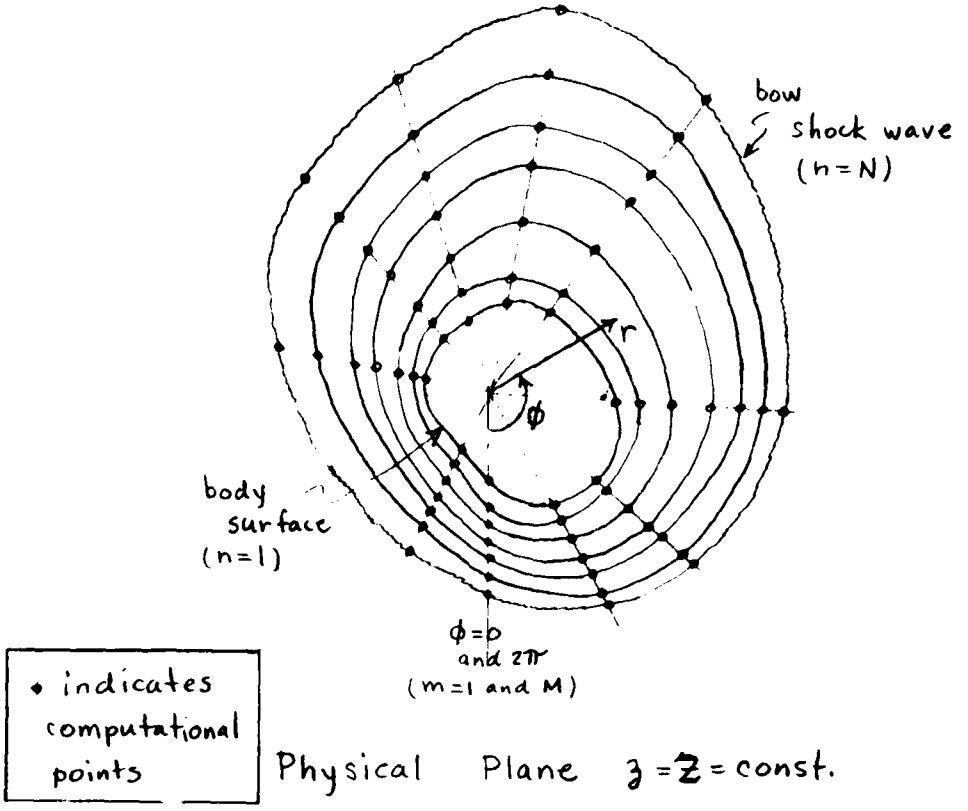


Physical Plane $z = Z = \text{const.}$



Computational Plane $Z = z = \text{const.}$

Fig. 2. Computational and corresponding physical plane for symmetric problem



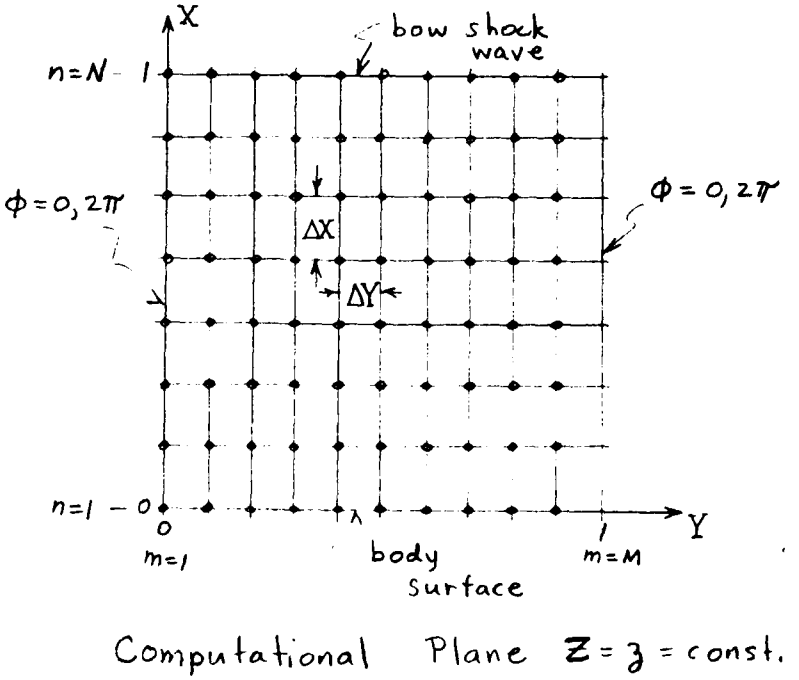


Fig. 3. Computational and corresponding physical plane for the non-symmetrical problem

physical planes for the symmetric and non-symmetric problems, respectively. As indicated in the figures, there are four types of points each requiring differing numerical procedures; interior points corresponding to $n=2,\cdots,N-1$; $m=2,\cdots,M-1$; points on the symmetry or periodic boundary planes $Y=0$ and $Y=1$ ($m=1$ and $m=M$, respectively); boundary points at the bow shock, $X=1$ ($n=N$); boundary points at the body surface, $X=0$ ($n=1$). For all points, the solution is advanced using predictor-corrector finite difference methods, i.e., the known solution at Z, say, is used to determine temporary (i.e., predicted) values at $Z + \Delta Z$; then the predicted values are used to determine the final solution (i.e., corrected) values at $Z + \Delta Z$.

In section 3.2 - 3.5 below, the particular method used for each of the above four types of points will be described. We shall assume that the quantities $\rho,u,v,w,p,c, c_\phi, c_z$ are known for $Z = Z^k$ on the mesh defined by (3.4). The objective is to determine these quantities on this mesh for $Z = Z^{k+1} = Z^k + \Delta Z$.

3.2 Interior Points

The numerical solution for all interior points is obtained by approximating the system (3.1) by the second-order accurate finite difference scheme of MacCormack (ref. 2). In the code, either of two MacCormack schemes can be selected. These are given by ($j = 0$ or 1):

$$\left.\begin{aligned} &\text{(predictor)}\quad U^{*}_{n,m} = U^{k}_{n,m} + \Delta Z\left(\frac{\partial U}{\partial Z}\right)^{k}_{n,m} \\ &\text{where} \\ &\left(\frac{\partial U}{\partial Z}\right)^{k}_{n,m} = -\left(\frac{F^{k}_{n+j,m} - F^{k}_{n+j-1,m}}{\Delta X}\right) - \left(\frac{G^{k}_{n,m+1} - G^{k}_{n,m}}{\Delta Y}\right) - E^{k}_{n,m} \end{aligned}\right\} \quad (3.6a)$$

$$\left.\begin{aligned} &\text{(corrector)}\quad U^{k+1}_{n,m} = \frac{1}{2}\left[U^{k}_{n,m} + U^{*}_{n,m} + \Delta Z\left(\frac{\partial U}{\partial Z}\right)^{*}_{n,m}\right] \\ &\text{where} \\ &\left(\frac{\partial U}{\partial Z}\right)^{*}_{n,m} = -\left(\frac{F^{*}_{n-j+1,m} - F^{*}_{n-j,m}}{\Delta X}\right) - \left(\frac{G^{*}_{n,m} - G^{*}_{n,m-1}}{\Delta Y}\right) - E^{*}_{n,m} \end{aligned}\right\} \quad (3.6b)$$

In the above, $U^{k}_{n,m} = U(Z^{k},X_{n},Y_{m})$. The quantities $F^{k}_{n,m}$, $G^{k}_{n,m}$, $E^{k}_{n,m}$ are evaluated from (3.3a) - (3.3c) at the point (Z^{k},X_{n},Y_{m}) using the known quantities ρ,u,v,w,p,c, and c_{z} with c_{ϕ} given by (3.5). The quantities $U^{*}_{n,m}$ are the predicted values of U at (Z^{k+1},X_{n},Y_{m}) from which predicted values of the flow quantities ρ,u,v,w,p can be determined using the definicions (2.1a), (2.2), and the thermodynamic relations. (This is described in the next paragraph). The quantities $F^{*}_{n,m}$, $G^{*}_{n,m}$, $F^{*}_{n,m}$ are evaluated from (3.3a) - (3.3c) at the point (Z^{k+1},X_{n},Y_{m}) using: the predicted values of ρ,u,v,w,p, c_{ϕ}, c_{z}, and the corrected values of c. The predictor and corrector procedure for c, c_{ϕ}, c_{z} will be described in the discussion of the bow shock points.

In the above, the flow variables ρ,u,v,w,p must be determined (or decoded) from the computed conservation variables U_{1},U_{2},U_{3},U_{4} after both the predictor and corrector steps. For decoding, it is convenient to introduce an effective gamma defined by

$$\Gamma = 1/\left(1 - \frac{p}{\rho h}\right) . \quad (3.7)$$

From the definitions (2.1a), we have

$$\left.\begin{aligned} u &= U_3/U_1 \ , \quad v = U_4/U_1 \\ p &= U_2 - U_1 w \ , \quad \rho = U_1/w \end{aligned}\right\} \tag{3.8a}$$

Substitution of (3.8a) into (2.2) yields

$$h = \frac{1}{2}\,[H_o - w^2] \ , \quad H_o = 2H_\infty - \frac{U_3^{\,2} + U_4^{\,2}}{U_1^{\,2}} \ . \tag{3.8b}$$

Substituting ρ and p given by (3.8a), and h given by (3.8b) into (3.7), we obtain a quadratic equation for w. The root of this equation corresponding to $w^2 > \Gamma p/\rho$ is

$$w = \frac{U_2\,[\Gamma + \sqrt{1 - \Phi}]}{U_1\,(\Gamma + 1)} \ , \quad \Phi = (\Gamma^2 - 1)[H_o\,(\frac{U_1}{U_2})^2 - 1] \tag{3.8c}$$

In the case of a perfect gas, $\Gamma = \gamma$, which is known, and equations (3.8c) and (3.8a) provide the desired decoding formulas. For a more general gas where $h = h(p,\rho)$ is given as a curve fit to thermodynamic data, Γ is not known a-priori and the decoding cannot be performed in closed form. In this case the decoding can be performed by solving the nonlinear equation

$$h(p,\rho) - \frac{1}{2}\,(H_o - w^2) = 0$$

for w. In this equation $h(p,\rho)$ is the value of h obtained from the curve fits with p and ρ defined by (3.8a). In the code, this equation is solved iteratively using the secant method. To start the iteration in the predictor cycle, the initial guess for w is obtained using (3.8c) with the value of Γ from the previous step; the second guess uses (3.8c) with the value of Γ defined from (3.7) which results from the initial trial.

For the corrector cycle, the iteration is started similarly except the initial guess uses in (3.8c) the value of Γ defined in the predictor step.

3.3 Bow Shock Points

At the bow shock boundary, the Rankine-Hugoniot relations (2.7) must be satisfied. These relations give the flow variables at the shock in terms of the shock geometry c, c_ϕ, and c_z (which are unknowns) and the free stream quantities. In the present procedure, a special system of equations for c, c_ϕ, and c_z is numerically solved using a second-order accurate predictor-corrector method to advance the shock geometry. This procedure differs from the more common practice of determining c_ϕ using central differences of c (see, refs 1,3,6,7,8, and 9).

The system of equations used for advancing c, c_ϕ, and c_z is developed in Appendix A (see, (A-20) - (A-22)). The resulting system is,

$$\frac{\partial c}{\partial Z} = c_z - (Y_z/Y_\phi)\, c_\phi \tag{3.9a}$$

$$\frac{\partial c_\phi}{\partial Z} = Y_\phi \frac{\partial c_z}{\partial Y} - Y_z \frac{\partial c_\phi}{\partial Y} \tag{3.9b}$$

$$\frac{\partial c_z}{\partial Z} = \frac{1}{C_1}\left\{\mathcal{R}_s - \frac{C_2}{c}\left[\left(Y_\phi \frac{\partial c_z}{\partial Y} - Y_z \frac{\partial c_\phi}{\partial Y}\right) - \frac{c_\phi}{c}\left(c_z - Y_z\, c_\phi/Y_\phi\right)\right]\right\} \tag{3.9c}$$

where

$$C_1 = \{(\nu_s\, w_\infty - c_z\, V_{n_\infty})\, A_1 - (p - p_\infty)[1 + (c_\phi/c)^2]\}/\nu_s^2$$

$$C_2 = \{[\nu_s\, v_\infty - (c_\phi/c)V_{n_\infty}]\, A_1 + c_z(c_\phi/c)(p - p_\infty)\}/\nu_s^2$$

$$A_1 = [\beta_o \rho_\infty V_{n_\infty} + \rho(\nu_s w_\infty - c_z V_{n_\infty})] A_o + \beta_o \rho_\infty (V_{n_\infty} - V_n)$$

$$A_0 = \frac{(V_{n_\infty} - V_n)[a^2 + V_n^2 + \mathcal{K}_1 (a^2/\rho)V_n(V_{n_\infty} - V_n)]}{V_{n_\infty}(a^2 - V_n^2)}$$

$$\beta_0 = \frac{1}{a}\sqrt{(w^2 - a^2)[1 + (c_\phi/c)^2] + [u - (c_\phi/c)v]^2}$$

$$\mathcal{K}_1 = (\frac{\partial \rho}{\partial h})_p = 1/(\frac{\partial h}{\partial \rho})_p$$

$$\mathcal{R}_s = \zeta \cdot (\frac{\partial F}{\partial X} + \frac{\partial G}{\partial Y} + E) - (A_1 - \nu_s \rho w)[v_\infty + (c_\phi/c)u_\infty]Y_z/(Y_\phi \nu_s)$$

$$\zeta = (\zeta_1, \zeta_2, \zeta_3, \zeta_4)$$

$$\zeta_1 = [2 - (\mathcal{K}_1/\rho)V^2]\lambda_-, \quad \zeta_2 = [(u - (c_\phi/c)v) - \lambda_-]/w + \lambda_- \mathcal{K}_1 w/\rho$$

$$\zeta_3 = \lambda_- \mathcal{K}_1 u/\rho - 1, \quad \zeta_4 = c_\phi/c + \lambda_- \mathcal{K}_1 v/\rho$$

$$\lambda_- = -a^2\{\beta_o w + [u - (c_\phi/c)v]\}/(w^2 - a^2)$$

In the above, the quantity $\zeta \cdot (\frac{\partial F}{\partial X} + \frac{\partial G}{\partial Y} + E)$ denotes the inner product of these vectors. Note that for a perfect gas

$$\mathcal{K}_1 = -\rho/h \quad [\text{perf}].$$

In the case of a more general gas where $h = h(p,\rho)$ is given as a curve fit to thermodynamic data, $\mathcal{K}_1$ is determined numerically using central differences.

The quantities c, c_ϕ, and c_z are advanced using predictor-corrector methods in the form:

$$\left.\begin{aligned} c_m^* &= c_m^k + Z\,(\frac{\partial c}{\partial Z})_{N,m}^k \\ (c_\phi)_m^* &= (c_\phi)_m^k + \Delta Z\,(\frac{\partial c_\phi}{\partial Z})_{N,m}^k \\ (c_z)_m^* &= (c_z)_m^k + \Delta Z\,(\frac{\partial c_z}{\partial Z})_{N,m}^k \end{aligned}\right\} \quad \text{(predictor)} \qquad (3.10)$$

$$\left.\begin{aligned} c_m^{k+1} &= \frac{1}{2}\,[c_m^k + c_m^* + \Delta Z\,(\frac{\partial c}{\partial Z})_{N,m}^*] \\ (c_\phi)_m^{k+1} &= \frac{1}{2}\,[(c_\phi)_m^k + (c_\phi)_m^* + \Delta Z\,(\frac{\partial c_\phi}{\partial Z})_{N,m}^*] \\ (c_z)_m^{k+1} &= \frac{1}{2}\,[(c_z)_m^k + (c_z)_m^* + \Delta Z\,(\frac{\partial c_z}{\partial Z})_{N,m}^*] \end{aligned}\right\} \quad \text{(corrector)} \qquad (3.11)$$

where in both the predictor and corrector steps the derivatives $\frac{\partial c}{\partial Z}$, $\frac{\partial c_\phi}{\partial Z}$, and $\frac{\partial c_z}{\partial Z}$ are determined using eqs. (3.9a) - (3.9c). In the predictor step, (3.9a) - (3.9c) are evaluated at $(Z^k,\ 1,\ Y_m)$ with the Y-derivatives approximated by

$$(\frac{\partial}{\partial Y})_{N,m}^k \approx \frac{(\quad)_{N,m+1}^k - (\quad)_{N,m}^k}{\Delta Y}$$

and the X derivative, $\frac{\partial F}{\partial X}$, approximated by

$$(\frac{\partial F}{\partial X})_{N,m}^k \approx \frac{F_{N,m}^k - F_{N-1,m}^k}{\Delta X}\ .$$

In the corrector step, (3.9a) - (3.9c) are evaluated at $(Z^k + \Delta Z,\ 1\ ,Y_m)$ using the predicted values of the flow variables and shock geometry (corrected value of c, see below) with the Y derivatives approximated by

$$(\frac{\partial}{\partial Y})_{N,m}^* \approx \frac{(\quad)_{N,m}^* - (\quad)_{N,m-1}^*}{\Delta Y}$$

and $\frac{\partial F}{\partial X}$ approximated by

$$\left(\frac{\partial F}{\partial X}\right)^{*}_{N,m} \approx \frac{F^{*}_{N,m} - F^{*}_{N-1,m}}{\Delta X} + \frac{F^{k}_{N,m} - 2F^{k}_{N,m-1} + F^{k}_{N,m-2}}{\Delta X}$$

The latter formula is used to achieve second order accuracy. The corrector for the shock shape c is performed before the corrector for c_ϕ, and c_z. This allows the use of corrected values of c in the evaluation of $\frac{\partial c_\phi}{\partial Z}$ and $\frac{\partial c_z}{\partial Z}$ in the corrector step and also in the Rankine-Hugoniot relations.

After advancing c, c_ϕ, and c_z in the predictor (or corrector), the predicted (or corrected) values of the flow variables at X = 1 are obtained from the Rankine-Hugoniot relations. These are rewritten for the purpose of the computation using the effective gamma defined by (3.7). The result is

$$\left.\begin{aligned} p &= \frac{1}{\Gamma+1}\left[p_\infty + \rho_\infty V_{n_\infty}^2 + \sqrt{(\rho_\infty V_{n_\infty}^2 - p_\infty\Gamma)^2 - \frac{2\rho_\infty V_{n_\infty}^2 (\Gamma-1)(\Gamma_\infty-\Gamma)}{\Gamma_\infty-1}}\right] \\ \rho &= \rho_\infty^2 V_{n_\infty}^2 / (\rho_\infty V_{n_\infty}^2 + p_\infty - p) \\ u &= u_\infty + \left(\frac{V_{n_\infty}}{\nu_s}\right)\left(1 - \frac{\rho_\infty}{\rho}\right),\ w = w_\infty - (u - u_\infty)c_z,\ v = v_\infty - (u - u_\infty)(c_\phi/c) \end{aligned}\right\} \quad (3.12)$$

In the above, V_{n_∞} is the free stream velocity component normal to the shock and $w_\infty, v_\infty, u_\infty$ are the free stream velocity components given by $V_\infty \cos\beta \cos\alpha$, $V_\infty (\cos\beta \sin\alpha \sin\phi - \sin\beta \cos\phi)$, and $-V_\infty (\cos\beta \sin\alpha \cos\phi + \sin\beta \sin\phi)$, respectively. For the case of a perfect gas, $\Gamma = \Gamma_\infty = \gamma$, and (3.12) give directly the flow quantities at X = 1 since V_{n_∞} is known when c, c_ϕ, and c_z are determined. For a more general gas where $h = h(p,\rho)$

is given as a curve fit, the Rankine-Hugoniot relation must be solved numerically. This is done by solving the nonlinear equation

$$h(p,\rho) - h_\infty - \frac{1}{2}(V_{n_\infty}^2 - V_n^2) = 0$$

for V_n. Here $h(p,\rho)$ is the value of h from the curve fits with p and ρ defined using the first and third equations in (2.7). This equation is solved iteratively using the secant method. To start the iteration, the required two estimates of V_n are obtained using

$$V_n = \rho_\infty V_{n_\infty}/\rho.$$

For the first estimate, ρ is the value obtained from (3.12) with Γ the previous value in the predictor step and the predicted value in the corrector step. For the second estimate, the value of Γ resulting from the first estimate is used.

3.4 Body Surface Points

At the body surface, X = 0 (n = 1), the boundary condition (2.6) must be satisfied. Our computational method for the points on X = 0 utilizes a special system of equations which is valid only on the body surface. These auxiliary equations are developed in Appendix A (see (A-10), (A-11), and (A-13) of Appendix A). Before giving the completely expanded forms of these equations, we will discuss the basic computational method.

The basic variables on X = 0 are the natural log of the pressure, $P = \log p$, the quantity $V_2 = v + (b_\phi/b)u$, and the entropy, s. These quantities are advanced on X = 0 using predictor-corrector methods in the form:

$$\left.\begin{aligned} P^*_{1,m} &= P^k_{1,m} + \Delta Z\left(\frac{1}{p}\frac{\partial p}{\partial Z}\right)^k_{1,m} \\ (V_2)^*_{1,m} &= (V_2)^k_{1,m} + \Delta Z\left(\frac{\partial V_2}{\partial Z}\right)^k_{1,m} \\ s^*_{1,m} &= s^k_{1,m} + \Delta Z\left(\frac{\partial s}{\partial Z}\right)^k_{1,m} \end{aligned}\right\} \text{(predictor)} \qquad (3.13)$$

$$\left.\begin{aligned} P^{k+1}_{1,m} &= \frac{1}{2}\left[P^k_{1,m} + P^*_{1,m} + \Delta Z\left(\frac{1}{p}\frac{\partial p}{\partial Z}\right)^*_{1,m}\right] \\ (V_2)^{k+1}_{1,m} &= \frac{1}{2}\left[(V_2)^k_{1,m} + (V_2)^*_{1,m} + \Delta Z\left(\frac{\partial V_2}{\partial Z}\right)^*_{1,m}\right] \\ s^{k+1}_{1,m} &= \frac{1}{2}\left[s^k_{1,m} + s^*_{1,m} + \Delta Z\left(\frac{\partial s}{\partial Z}\right)^*_{1,m}\right] \end{aligned}\right\} \text{(corrector)} \qquad (3.14)$$

where in both the predictor and corrector steps the derivatives $\frac{\partial p}{\partial Z}$, $\frac{\partial V_2}{\partial Z}$, and $\frac{\partial s}{\partial Z}$ are determined from eqs. (A-10), (A-11), and (A-13).

(The completely expanded form of these equations will be presented below in this section). After both the predictor and corrector steps, the flow variables ρ,u,v,w,p are determined as follows. From the computed value of P, we obtain the pressure; i.e., $p = \exp(P)$. Using thermodynamic relations, we obtain ρ and h from p and s. It follows from (2.2) that $V^2 = u^2 + w^2 + v^2 = 2(H_\infty - h)$ and from (2.6), we have

$$\left.\begin{aligned} w &= \sqrt{[1+(b_\phi/b)^2]V^2 - V_2^2} \;/\; \nu_w, \\ \nu_w^2 &= 1+b_z^2+(b_\phi/b)^2 \\ v &= [V_2 - (b_\phi/b)b_z w]/[1 + (b_\phi/b)^2], \\ u &= b_z w + (b_\phi/b)v \end{aligned}\right\} \qquad (3.15)$$

Note, the boundary condition (2.6) is satisfied by both predicted and corrected values of the flow variables.

The above formulation, of identifying the variables P, V_2, and s as the variables to be advanced at the boundary, provides a form of the equations which allows one to maintain an accurate approximation through regions where the flow nonuniformity develops and remains adjacent to the body surface (e.g. entropy layers on blunted slender bodies). Other formulations which utilize finite differencing in the vicinity of such flow non-uniformities characteristically are differencing nonsmooth flow quantities and hence require some local modifications to maintain the calculation. However, our formulation allows one to advance the

solution by the differencing of only smooth quantities. In order to see why this is so, first observe that, since such nonuniformities lie along the body surface which is a stream surface, they have the character of a contact discontinuity. It is known (c.f., ref. 4, p. 317-18) that across such a discontinuity only the pressure and the normal component of velocity can never experience jumps. Now consider the present wall point computation. The only X differences required are in (A-10), the equation for pressure. The quantities V_2 and s are advanced using only quantities defined on the wall. Further, the only quantities differenced in the X direction are p and A. But, p is smooth across the nonuniformity and A at the wall is, except for a factor involving geometry, the normal component of velocity which is also smooth across the nonuniformity. Hence, the wall point calculation remains well behaved. Consider now the calculation for the adjacent interior point, $X = \Delta X$. In either the predictor or corrector (depending on j=0 or 1) an X difference of F must be taken across the nonuniformity. But at X = 0, F involves only the pressure and geometry (since A = 0 at the wall in both the predictor and the corrector). Thus, in the calculation at $X = \Delta X$, the X difference of F is unaffected by the nonuniformity.

The formulas used for determining $\frac{\partial p}{\partial Z}$, $\frac{\partial V_2}{\partial Z}$, and $\frac{\partial s}{\partial Z}$ in (3.13) and (3.14) are (A-10), (A-11), and (A-13), respectively. The actual formulas used in the code are obtained by expanding these equations using the boundary condition (2.6) and noting that at X = 0, $X_z = - b_z X_r$ and $X_\phi = - b_\phi X_r$. From (A-10), we obtain

$$\frac{\partial p}{\partial Z} = X_r \lambda_+ \frac{\partial p}{\partial X} - \frac{1}{\beta_1}\{\rho w[\lambda_+ \frac{\partial A}{\partial X} - (a_7 w + a_4 v)] + \mathcal{P}\} \tag{3.16}$$

where

$$\mathcal{P} = \frac{\rho v}{b} V_2 + w\lambda_+ e_1 + \xi \cdot \frac{\partial G}{\partial Y}$$

$$+ p[\xi_2(T_{g_5} Y_z + T_{g_6}) + \frac{1}{b}\xi_4(T_{g_5} Y_\phi + b_\phi/b)] \qquad (3.16a)$$

$$\lambda_+ = \frac{a^2(\beta_1 - b_z)}{w^2 - a^2}, \quad \beta_1 = \sqrt{\left(\frac{w^2}{a^2}\right)\nu_w^2 - [1 + (\frac{b_\phi}{b})^2]}$$

$$a_7 = \frac{\partial b_z}{\partial Z} = b_{zz} - b_{z\phi} Y_z / Y_\phi$$

$$a_4 = \frac{\partial (b_\phi/b)}{\partial Z} = \frac{1}{b}[b_{z\phi} - \frac{b_z b_\phi}{b} - (b_{\phi\phi} - b_\phi^2/b) Y_z / Y_\phi]$$

$$\xi = (\xi_1, \xi_2, \xi_3, \xi_4)$$

$$\xi_1 = w\lambda_+(2 - V^2 \mathcal{K}_1/\rho), \quad \xi_2 = b_z - \lambda_+ + w^2 \mathcal{K}_1 \lambda_+/\rho$$

$$\xi_3 = wu\mathcal{K}_1\lambda_+/\rho - 1, \quad \xi_4 = wv\mathcal{K}_1\lambda_+/\rho + b_\phi/b$$

$$\mathcal{K}_1 = (\frac{\partial \rho}{\partial h})_p = 1/(\frac{\partial h}{\partial \rho})_p$$

In the above, the quantity $\xi \cdot \frac{\partial G}{\partial Y}$ denotes the inner product of these vectors. Note that for a perfect gas $\mathcal{K}_1 = -\rho/h$; **for a more general gas $\mathcal{K}_1$ is determined numerically using central differences in the curve fits.**

From (A-11), we obtain

$$\frac{\partial V_2}{\partial Z} = a_4 u + \mathcal{V}/\rho w \qquad (3.17)$$

where

$$\mathcal{V} = \eta \cdot \frac{\partial G}{\partial Y} - \rho v w b_z/b - \frac{p}{b}(T_{g_5} Y_\phi + b_\phi/b) \tag{3.17a}$$

$$\eta = (\eta_1, \eta_2, \eta_3, \eta_4)$$

$$\eta_1 = V_2 \quad , \quad \eta_2 = 0 \quad , \quad \eta_3 = - b_\phi/b \quad , \quad \eta_4 = -1$$

Equation (A-13) remains

$$\frac{\partial s}{\partial Z} = - \frac{B}{w} \frac{\partial s}{\partial Y} \, . \tag{3.18}$$

In the predictor step (i.e., (3.13)), the quantities $(\frac{\partial p}{\partial Z})_{1,m}$, $(\frac{\partial V_2}{\partial Z})_{1,m}$, and $(\frac{\partial s}{\partial Z})_{1,m}$ are obtained by evaluating (3.16) - (3.18) at $(Z^k, 0, Y_m)$ with the X partial derivatives replaced by

$$(\frac{\partial}{\partial X})^k_{1,m} = \frac{(\,)^k_{2,m} - (\,)^k_{1,m}}{\Delta X}$$

and the Y partial derivatives replaced by

$$(\frac{\partial}{\partial Y})^k_{1,m} = \frac{(\,)^k_{1,m+1} - (\,)^k_{1,m}}{\Delta Y} \, .$$

For the corrector step (3.16) - (3.18) are evaluated at $(Z^k + \Delta Z, 0, Y_m)$ using predicted values of the flow variables with the X partial derivatives replaced by

$$(\frac{\partial}{\partial X})^*_{1,m} = \frac{(\,)^*_{2,m} - (\,)^*_{1,m}}{\Delta X}$$

and the Y partial derivatives replaced by

$$(\frac{\partial}{\partial Y})^*_{1,m} = \frac{(\,)^*_{1,m} - (\,)^*_{1,m-1}}{\Delta Y} \, .$$

An alternative proceedure for the wall point calculation can be obtained by rewritting (3.16a) and (3.17a) using the definitions of G and e_1, (2.6) and (2.2), and the thermodynamic relation (A-12), i.e.,

$$\mathcal{P} = \frac{\rho v}{b} V_2 + [b_z Y_z + \frac{1}{b^2} b_\phi Y_\phi + (\frac{Bw}{a^2} - Y_z) \lambda_+] \frac{\partial p}{\partial Y}$$

$$- \rho B \quad (a_5 w + a_3 v) \tag{3.16b}$$

$$+ \rho w \lambda_+ \{ (T_{f_6} - T_{g_6}) w + \frac{v}{b} T_{f_7} + \frac{w}{b} [Y_\phi \frac{\partial (v/w)}{\partial Y} + b_z] \}$$

where

$$a_5 = \frac{\partial b_Z}{\partial Y} = b_{Z\phi}/Y_\phi$$

$$a_3 = \frac{(b_\phi/b)}{\partial Y} = [b_{\phi\phi}/b - (b_\phi/b)^2]/Y_\phi$$

and

$$\mathcal{V} = \rho B(a_3 u - \frac{\partial V_2}{\partial Y}) - \frac{1}{b} Y_\phi \frac{\partial p}{\partial Y} - \frac{\rho v w}{b} b_z \tag{3.17b}$$

We have found that the use of (3.16b) and (3.17b) instead of (3.16a) and (3.17a) can in certain instances yield significantly different numerical results. The use of (3.16a) and (3.17a) appears to provide better results on the lee-side of bodies at large angle of attack; in fact, in sphere-cone calculations the use of (3.16a) and (3.17a) yields numerical results at large angles of attack where the calculation using (3.16b) and (3.17b) fails. On the other hand, immediately downstream of discontinuities in body slope which produce large expansions, the use of (3.16b) and (3.17b) produces numerical results where the calculation using (3.16a) and (3.17a) fails. The reason for these differing numerical behaviors, using essentially

equivalent analytical formulas, is not understood at the present time. In the present code, both formulations are available; either form can be selected by the user. The scheme using (3.16b) and (3.17b) is automatically used by the code after certain discontinuities in body slope are encountered (see section 4.1 for details).

In both formulations, the X partial derivatives appearing in (3.16) are forward differenced in both the predictor and corrector steps. This makes the numerical solution at the wall formally first order accurate. Both formulations can be made formally second order accurate (ref. 1) by adding a correction to the corrector formula for the pressure; viz.,

$$p^{k+1}_{1,m} = \frac{1}{2}\{p^{k}_{1,m} + p^{*}_{1,m} + \Delta Z[(\frac{1}{p}\frac{\partial p}{\partial Z})^{*}_{1,m} + (\frac{1}{p^{*}_{1,m}})\,\tilde{\rho}^{k}_{1,m}]\} \tag{3.19}$$

where

$$\tilde{\rho}^{k}_{1,m} = (\lambda_{+})^{k}_{1,m}\,[(X_r)^{k}_{1,m}(\frac{2p^{k}_{2,m} - p^{k}_{3,m} - p^{k}_{1,m}}{\Delta X}) - (\rho w/\beta_1)^{k}_{1,m}\,(\frac{2A^{k}_{2,m} - A^{k}_{3,m}}{\Delta X})]$$

The term $\tilde{\rho}^{k}_{1,m}$ is computed in the predictor step, using values at $(Z^{k},0,Y_{m})$, and applied in the corrector step.

In the present code, second order accuracy is an option which can be selected by the user. However, care must be exercised in the use of this option. The higher accuracy can be achieved only when the computed flow field near the wall is sufficiently smooth. In certain instances,

specifically downstream of discontinuities in body slope and/or body curvature, numerical oscillations will occur in the computed wall pressure when (3.19) is used. The magnitude of these oscillations depends on the size of the discontinuity; the case of body slope discontinuities is more serious--many times causing the calculation to fail. The code automatically switches the second order accurate scheme off (if it was originally selected) when discontinuities in body slope are encountered (see section 4.1 for details). Note that when the only flow nonuniformity present is of the type lying along the wall (e.g., an entropy layer), the second order scheme can still be used. This follows because the only quantities which are differenced in the X direction are p and A which are, as we have previously indicated, smooth quantities up to the wall.

3.5 Symmetry and Periodic Boundary Points

The procedures given above for wall, interior, and shock points must be slightly modified at the boundary planes Y=0 and Y=1 (corresponding to m=1 and M, respectively) since some of the differences in the Y direction require quantities defined at $Y = -\Delta Y$ and $Y = 1 + \Delta Y$. The required quantities are obtained using either symmetry or periodic conditions depending on whether the symmetric or the non-symmetric problem is being considered. Additional considerations come into play when clustering transformations are being used. For computational simplicity certain restrictions are placed on the mapping functions f(X,Y,Z) and g(Y,Z). These restrictions vary depending on whether the symmetric or the non-symmetric problem is being considered.

Consider first the symmetric problem. Here the planes $\phi=0$ and $\phi=\pi$ are boundaries where all flow variables are even functions of ϕ except v which is an odd function of ϕ. Since these conditions must be applied in the computational space (X,Y,Z), it is natural to impose on the mapping function f(X,Y,Z), the condition

$$f_Y(X,0,Z) = f_Y(X,1,Z) = 0 \tag{3.20}$$

Now, given q_e, an even function of ϕ about $\phi=0$ and π in the physical plane, it can be shown using (3.1), (3.2), when (3.20) is satisfied, that in the computational plane

$$\left.\begin{aligned} q_e(X,-\Delta Y,Z) &= q_e(X,\Delta Y,Z) + 0(\Delta Y^3) \\ \text{and} \\ q_e(X,1+\Delta Y,Z) &= q_e(X,1-\Delta Y,Z) + 0(\Delta Y^3) \end{aligned}\right\} \tag{3.21}*$$

further, if q_o is an odd function of ϕ about 0 and π then

$$\left.\begin{aligned} q_o(X,-\Delta Y,Z) &= -\, q_o(X,\Delta Y,Z)D(0,Z) + 0(\Delta Y^4) \\ \text{and} \\ q_o(X,1+\Delta Y,Z) &= -\, q_o(X,1-\Delta Y,Z)/D(1,Z) + 0(\Delta Y^4) \end{aligned}\right\} \tag{3.22}*$$

where

$$D = [2-(g_{YY}/g_Y)\Delta Y]/[2+(g_{YY}/g_Y)\Delta Y] \ .$$

In the computation for the symmetric problem, the points on Y=0 and Y = 1 + ΔY are now treated in essentially the same manner as discussed in the previous sections. Except for the quantities needed on the "fringe" planes Y = - ΔY and Y = 1 + ΔY. These are determined using either (3.21) or (3.22). The derivatives of g are determined by direct evaluation of their analytical definitions at Y = - ΔY and 1 + ΔY. Thus, it is required

*The derivation of these expressions is carried out in Appendix B.

that $g(Y,Z)$ be defined (and twice continuously differentiable) for $-\Delta Y \leq Y \leq 1+\Delta Y$. In both the predictor and corrector steps the boundary conditions

$$v(X,0,Z) = v(X,1,Z) = V_2(0,Z) = V_2(1,Z) = 0 \tag{3.23}$$

are imposed.

Consider now the non-symmetric problem; i.e., all quantities are periodic functions of ϕ with period 2π in the physical plane. It is convenient for computational purposes not to destroy this periodicity in the computational space; i.e., to have all quantities periodic functions of Y with period 1. This can be achieved by requiring that the mapping function $f(X,Y,Z)$ be periodic in Y with period 1 and that the mapping function $g(Y,Z)$ be such that $[g(Y,Z)-Y]$ is a periodic function of Y with period 1. With these conditions on $f(X,Y,Z)$ and $g(Y,Z)$, all quantities are periodic in the computational space. The calculation is now performed for only M-1 planes ($m=1,2,\cdots M-1$) corresponding to $0 \leq Y \leq 1-\Delta Y$. For Y=0 (m=1) and $Y=1-\Delta Y$ (m=M-1) it will be required, in either the predictor or corrector, to provide quantities on $Y=-\Delta Y$ and Y=1, respectively. These are determined by the periodicity conditions

$$\left.\begin{aligned} q(X,-\Delta Y,Z) &= q(X,1-\Delta Y,Z) \\ \text{and} \\ q(X,1,Z) &= q(X,0,Z) \end{aligned}\right\} \tag{3.24}$$

where q is any quantity.

3.6 Step Size and Stability

The step size ΔZ is chosen in accordance with the CFL condition (which is a necessary condition for numerical stability). This condition

is that the domain of dependence of the partial differential equations must be contained in the domain of dependence of the finite difference equations at all points. The computation of ΔZ based on this condition is carried out in Appendix C. We give only the final results here. Note that two finite difference schemes are available in the code (corresponding to j=0 or 1 in (3.6a) and (3.6b)) and the CFL condition for each scheme is different. The CFL conditions for the present code are given by

$$\Delta Z \le \Delta X \min\left\{\frac{w^2-a^2}{\mu}\right\}$$

where the minimum is taken over all computational points. At each point, μ is defined by

$$\mu = \max(\mu_1, \mu_2, \mu_3)$$

where

$$\mu_1 = |wA-a^2X_z| + a\sqrt{(w^2-a^2)(X_r^2 + X_\phi^2/r^2) + (A-wX_z)^2}$$

$$\mu_2 = |\delta|\left[|wB-a^2Y_z| + a\sqrt{(w^2+v^2-a^2)(Y_\phi^2/r^2)}\right]$$

$$\mu_3 = |wA-a^2X_z - \delta(wB-a^2Y_z)|$$

$$+ a\sqrt{(w^2-a^2)\left[X_r^2+\frac{1}{r^2}(X_\phi-\delta Y_\phi)^2\right] + (wX_z-A+\delta vY_\phi/r)^2}$$

and

$$\delta = \begin{cases} \Delta X/\Delta Y, & \text{if } j=0 \\ -\Delta X/\Delta Y, & \text{if } j=1 \end{cases}$$

In the code, ΔZ is computed using

$$\Delta Z = \mathcal{C}\,\Delta X \min\left\{\frac{w^2-a^2}{\mu}\right\} \tag{3.25}$$

where $0 < \mathcal{C} \le 1$ is a numerical constant which can be selected by the user. Certain adjustments in $\mathcal{C}$ can be made when discontinuities in body slope are encountered (see section 4.1 for details).

4. SPECIAL FEATURES

4.1 Discontinuities in Body Slope

Many important flow field applications involve body shapes given by functions $b(\phi,z)$ that contain discontinuous slopes b_z and b_ϕ, e.g., bodies which have slices and/or flaps and biconics or other segmented body shapes. It is well known that when such discontinuities appear certain discontinuities in the flow variables at the surface are produced. Furthermore, these flow discontinuities generally propagate into the flow field as either shock waves or expansion fans. Numerically, such discontinuities violate the assumptions which underlie the basic numerical algorithms. Computing the flow field by marching through these discontinuities without any special provisions produces numerical oscillations downstream of the discontinuity in both the flow variables on the body surface and in the interior of the flow. When the discontinuities are large, these oscillations can result in a program halt or, at the very least, a substantial region of unrealistic results.

The present code incorporates special provisions which are used when certain discontinuities in body slope are encountered. The code approximately locates these body discontinuities and then computes the associated flow discontinuities in the surface flow variables based on a local analysis of the discontinuity. We have found that this procedure gives, in most cases, far better results than would be obtained by simply marching through the discontinuity. The present procedure, however, is not by itself a complete solution to all the problems associated with body slope discontinuities. Even though the scheme for interior points is of the "shock capturing" type, numerical oscillations in the interior, downstream of a body

discontinuity, can develop when the discontinuity produces large expansion or compression discontinuities. This is a defect in the MacCormack scheme. In certain instances some numerical oscillations also occur along the wall immediately downstream of the body slope discontinuity. A consistent method for computing the body surface points immediately downstream of discontinuities of this type is not known at the present time. At the end of this section, after the discussion of the procedure for obtaining discontinuities in the surface flow variables, various techniques available in the code for improving the downstream calculation will be discussed.

For complicated geometries (e.g., bodies with slices), the precise location of body slope discontinuities would require a considerable amount of additional logic and computations. For this reason, in the present code the discontinuities are only located approximately; i.e., within at most ΔZ. The procedure is as follows. At every step $Z^{k+1} = Z^k + \Delta Z$, the values of b_z and b_ϕ are compared to their previous values (at Z^k) using

$$\left|(b_z)_m^k - (b_z)_m^{k+1}\right| - \Delta Z \max\left\{\left|(b_{zz})_m^k\right|, \left|(b_{zz})_m^{k+1}\right|\right\} > \epsilon \qquad (4.1)$$

$$\left|(b_\phi)_m^k - (b_\phi)_m^{k+1}\right| - \Delta Z \max\left\{\left|(b_{z\phi})_m^k\right|, \left|(b_{z\phi})_m^{k+1}\right|\right\} > \epsilon \qquad (4.2)$$

where ϵ is a small positive number (we use $\epsilon = 10^{-6}$).

When either of the above inequalities is satisfied, the code assumes that a discontinuity in b_z and/or b_ϕ exists between Z^k and Z^{k+1} at $Y = Y_m$. At each such value of Y, the body geometry is temporarily modified* at Z^{k+1} by putting the body shape derivatives b_z, b_ϕ, b_{zz}, $b_{z\phi}$, and $b_{\phi\phi}$

*The body that is in effect created at $Z = Z^{k+1}$ by this convention will be referred to as the "modified body."

equal to their values at Z^k. This temporarily removes the discontinuities in the body slopes and the predictor-corrector sequence for the "modified body" is performed for the step Z^{k+1} using the basic scheme as described in section 3. After the corrector step, the body derivatives which were modified are put equal to their true values at Z^{k+1}. Also, for each value of Y where discontinuities were sensed, the flow quantities at the body surface are changed by applying a local analysis to determine the appropriate flow discontinuities as if the body slope discontinuity were located at $Z = Z^{k+1}$. Note that this special procedure is only applied to discontinuities of b_z and b_ϕ in the Z direction. No special consideration is given to other discontinuities in body shape (e.g. discontinuities in b_ϕ for a fixed Z).

Let us now turn to the procedure for determining the appropriate flow discontinuities at the wall points $(0,Y_m,Z^{k+1})$ where according to the above procedure discontinuities in body slope have been effectively placed. The analysis is conveniently carried out in the physical space (r,ϕ,z). Let $(r_o,\phi_o,z_o)^+$ where $r_o = b(\phi_o,z_o)$ be the body surface point corresponding to $(0,Y_m,Z^{k+1})$. We will denote the values of quantities associated with the "modified body" geometry by the subscript - and the values of quantities associated with the true body geometry by the subscript +. The modified body geometry will be referred to as the upstream side of (r_o,ϕ_o,z_o); the true body geometry will be referred to as the downstream side of (r_o,ϕ_o,z_o)*. All flow quantities on the upstream side of (r_o,ϕ_o,z_o) are the computed

*This terminology is motivated by the fact that the flow on the surface of the modified body is in the direction of (r_o,ϕ_o,z_o).

†The symbol ϕ_o used in this section bears no connection to its use in other sections of the report.

values at this point using the modified geometry and thus are known. The problem is therefore to determine the flow quantities on the downstream side of (r_o,ϕ_o,z_o) given the upstream quantities and the body slopes on both sides.

Because b_z and/or b_ϕ have different values on the upstream and downstream sides of (r_o, ϕ_o,z_o), there are two distinct body normal vectors at this point; i.e.,

$$\vec{n}_{\mp} = - (b_\phi/b)_{\mp} \vec{e}_\phi + \vec{e}_r - (b_z)_{\mp} \vec{e}_z \tag{4.2}$$

The situation is depicted in Fig. 4. The vector $\vec{\tau} = \vec{n}_+ \times \vec{n}_-$ is tangent to the edge of the discontinuity at (r_o,ϕ_o,z_o); c.f., Fig. 4. We introduce the vectors $\vec{\sigma}_{\mp} = \vec{n}_{\mp} \times \vec{\tau}$ and consider the two sets of mutually perpendicular unit vectors, one for each side of (r_o,ϕ_o,z_o), given by

$$(\vec{e}_n)_{\mp} = \vec{n}_{\mp}/|\vec{n}_{\mp}|, \ \vec{e}_\tau = \vec{\tau}/|\vec{\tau}|, \ (\vec{e}_\sigma)_{\mp} = \vec{\sigma}_{\mp}/|\vec{\sigma}_{\mp}|. \tag{4.3}$$

With respect to the above notation, the basic conditions used to determine the flow quantities on the downstream side of (r_o,ϕ_o,z_o) are:

i.) $\vec{V}_- \cdot \vec{\tau} = \vec{V}_+ \cdot \vec{\tau}$; i.e., there is no change in the velocity component tangent to the edge.

ii.) The quantities p_+,ρ_+, and $[\vec{V}_+ - (\vec{V}_+ \cdot \vec{e}_\tau)\vec{e}_\tau]$ are to be determined so that the inviscid boundary condition $\vec{V}_+ \cdot \vec{n}_+ = 0$ is satisfied.

Note that, by computation, the upstream flow satisfies the boundary condition $\vec{V}_- \cdot \vec{n}_- = 0$. Also, using i.) and applying the condition $\vec{V}_+ \cdot \vec{n}_+ = 0$, we obtain

$$\vec{V}_+ = (\vec{V}_- \cdot \vec{e}_\tau)\vec{e}_\tau + V_{\sigma_+} \vec{e}_{\sigma_+} \tag{4.4}$$

where $V_{\sigma_+} = (\vec{V}_+ \cdot \vec{e}_{\sigma_+})$ is the only unknown quantity.

The actual procedure for carrying out ii.) depends on the value of $V_{\sigma_-} = \vec{V}_- \cdot \vec{e}_{\sigma_-}$, the known upstream component of the surface velocity normal

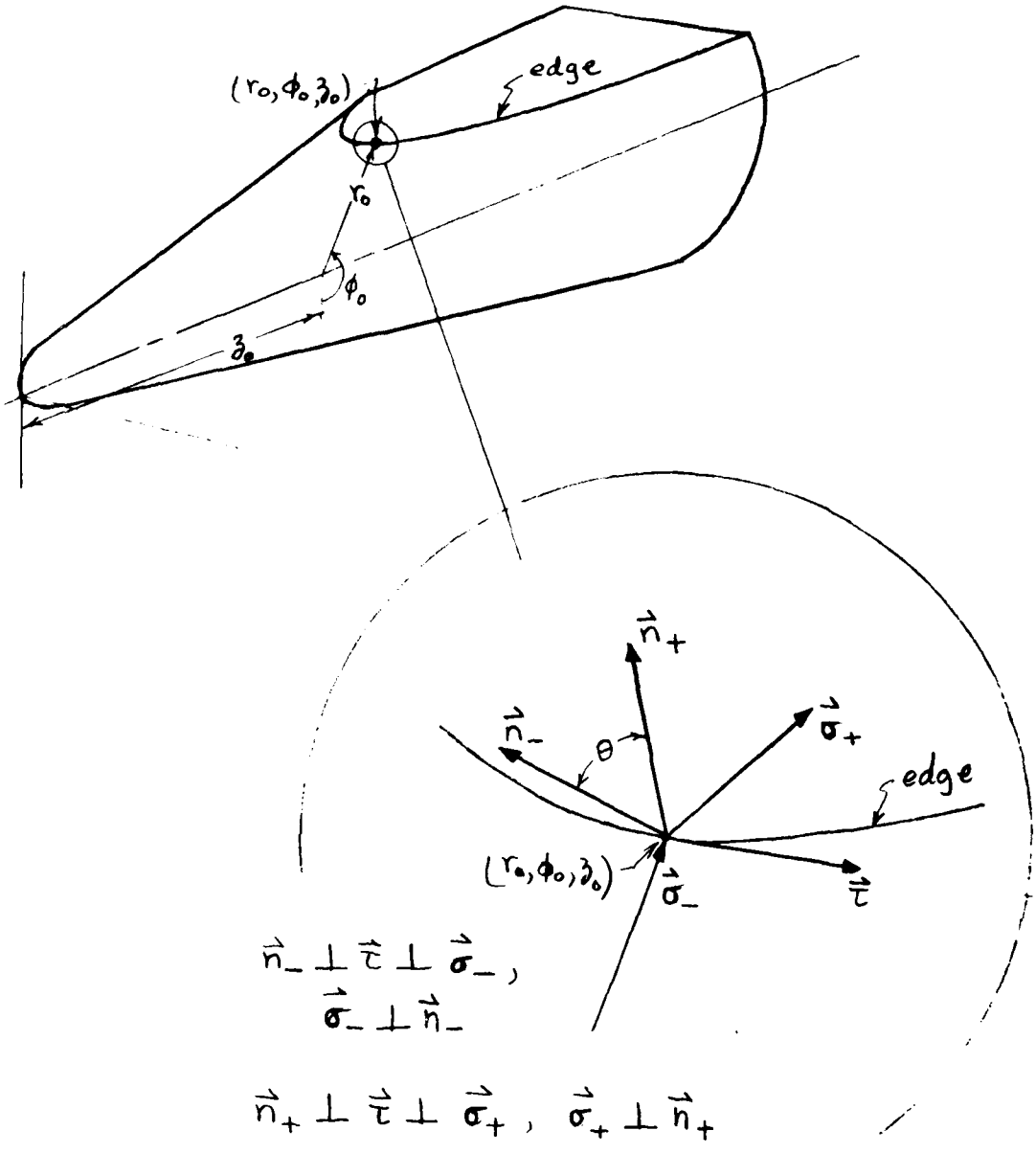


Fig. 4. Orthogonal directions used in the analysis of body slope discontinuities.

to the edge. From standard vector identities and $\vec{V}_- \cdot \vec{e}_{n_-} = 0$, it follows that

$$V_{\sigma_-} = |\vec{n}_-|(\vec{V}_- \cdot \vec{n}_+) \Big/ \sqrt{|\vec{n}_-|^2 |\vec{n}_+|^2 - (\vec{n}_- \cdot \vec{n}_+)^2} \tag{4.5}$$

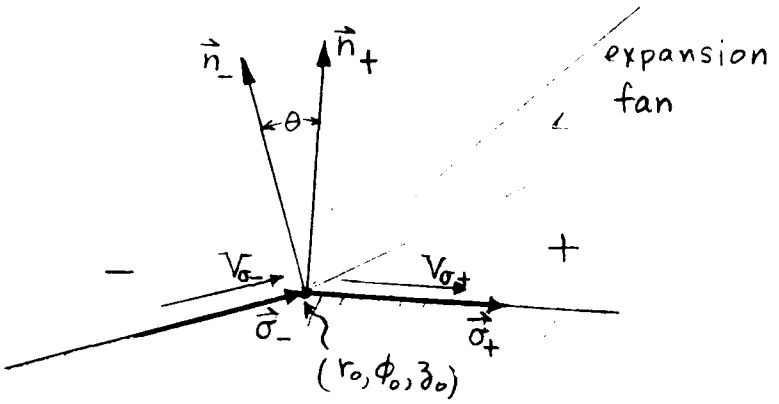
When $|V_{\sigma_-}| < a_-$, the flow across the edge is subsonic. In this case, a heuristic procedure is used; namely, the velocity vector is rotated to satisfy $\vec{V}_+ \cdot \vec{n}_+ = 0$ assuming there are no discontinuities in $|\vec{V}|$, p, and ρ; i.e.,

$$p_+ = p_-, \ \rho_+ = \rho_-, \text{ and } V_{\sigma_+} = V_{\sigma_-} . \tag{4.6}$$

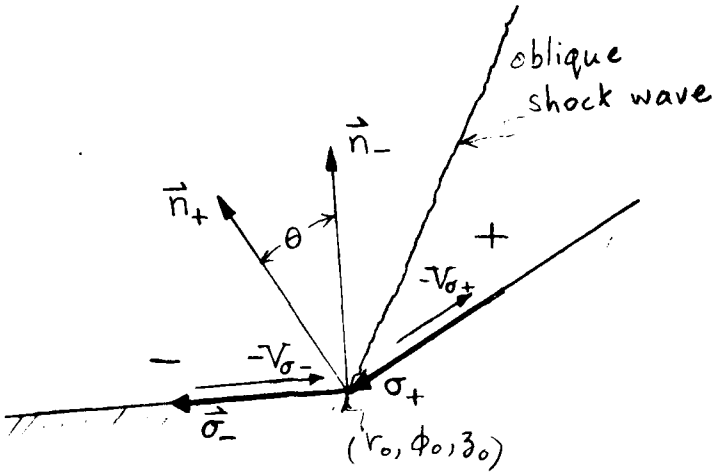
When $|V_{\sigma_-}| \geq a_-$, the flow across the edge is supersonic. This case has been analyzed by Prandtl and Meyer in a classical investigation (see; e.g., ref. 4, chapter XI). This analysis indicates that the quantities p_+, ρ_+, and V_{σ_+} can be determined using classical two-dimensional supersonic turning relations in the plane determined by $\vec{n}_+$ and $\vec{n}_-$. The angle through which the flow is turned is given by

$$\theta = \cos^{-1}[(\vec{n}_+ \cdot \vec{n}_-)/|\vec{n}_+||\vec{n}_-|] \tag{4.7}$$

There are two cases; viz., $V_{\sigma_-} > 0$ and $V_{\sigma_-} < 0$. When $V_{\sigma_-} > 0$, it follows from (4.5) that $(\vec{V}_- \cdot \vec{n}_+) > 0$ and thus the flow is turned by a centered expansion fan attached to the edge (see, Fig. 5a). When $V_{\sigma_-} < 0$, it follows that $(\vec{V}_- \cdot \vec{n}_+) < 0$ and the flow is turned by an oblique shock wave attached to the edge (see, Fig. 5b). In the latter case, it can happen that conditions are such that no oblique shock wave exists; e.g., this can occur when θ is large and/or $(V_{\sigma_-}/a_-)^2$ is near one. In this situation, the Prandtl-Meyer analysis breaks down and a heuristic procedure is used. This will be discussed later in this section.



(a.) $\vec{V}_- \cdot \vec{n}_+ > 0$, $V_{\sigma_-} > 0$



(b.) $\vec{V}_- \cdot \vec{n}_+ < 0$, $V_{\sigma_-} < 0$

Fig. 5. Flow discontinuities in the case $V_\sigma^2 > a^2$.

In the case when $V_{\sigma_-} > 0$, p_+, ρ_+, and V_{σ_+} are obtained from the equations for a centered expansion written in the following differential form;

$$\left.\begin{aligned} &\frac{dp}{d\alpha} = -\rho q^2 / \sqrt{q^2/a^2 - 1} \\ &\text{where} \\ &q^2 = 2H_\infty - (\vec{V}_- \cdot \vec{e}_\tau)^2 - 2h \\ &\rho = \rho(s_-, p),\ h = h(p,\rho),\ a^2 = a^2(p,\rho) \\ &s_- = s(p_-, \rho_-) \end{aligned}\right\} \quad (4.8)$$

In the code (4.8) is numerically integrated on the interval $0 \le \alpha \le \theta$ (where θ is given by (4.7)) subject to the initial condition $p = p_-$ at $\alpha = 0$ using the second order Improved-Euler method with 18 subdivisions. The final results at $\alpha = \theta$ give $p_+ = p_{\alpha=\theta}$, $V_{\sigma_+} = q_{\alpha=\theta}$, $\rho_+ = \rho_{\alpha=\theta}$.

In the case when $V_{\sigma_-} < 0$, p_+, ρ_+, and V_{σ_+} are obtained using the oblique shock relations. Consider first the case of a perfect gas. In this case, the shock relations are given in ref. 5 (pp. 9-10); they are reproduced here for the sake of completeness. Let $m^2 = (V_{\sigma_-}/a_-)^2$ and solve

$$\left.\begin{aligned} &\chi^3 + c_1\chi^2 + c_2\chi + c_3 = 0 \\ &c_1 = -(1 + 2/m^2) - \gamma \sin^2\theta \\ &c_2 = (2m^2 + 1)/m^4 + [(\gamma + 1)^2/4 + (\gamma - 1)/m^2]\sin^2\theta \\ &c_3 = \cos^2\theta / m^4 \end{aligned}\right\} \quad \begin{matrix}[\text{perf}] \\ (4.9)\end{matrix}$$

for the middle root (in the above, θ is the turning angle, see (4.7)). This value of χ is the sine-squared of the shock angle. We then have

$$\left.\begin{aligned} p_+/p_- &= [2\gamma \mathcal{M}^2\chi - (\gamma - 1)]/(\gamma + 1) \\ \rho_+/\rho_- &= (\gamma + 1)\mathcal{M}^2\chi/[(\gamma - 1)\mathcal{M}^2\chi + 2] \\ V_{\sigma_+}/V_{\sigma_-} &= \sqrt{1 - 4(\mathcal{M}^2\chi - 1)(\gamma\mathcal{M}^2\chi + 1)/[(\gamma + 1)^2\mathcal{M}^4\chi]} \end{aligned}\right\} \text{[perf]} \quad (4.10)$$

As we pointed out earlier, (4.9) may not yield a root which is physically meaningful (i.e., one that increases entropy) for certain values of $\mathcal{M}$ and θ. This occurs when

$$(2c_1^3 - 9c_1c_2 + 27c_3)^2 + 12(3c_3 - c_2^2)^3 > 0 .$$

In this special case, a heuristic procedure is used; namely, a normal shock wave (i.e., normal to $\vec{\sigma}_-$) is assumed to exist on the upstream side in front of (r_o,ϕ_o,z_o). The flow quantities behind this normal shock are obtained using (4.10) with $\chi = 1$. The final results for p_+, ρ_+, and V_{σ_+} are obtained by performing an isentropic expansion from the normal shock values to the turning angle θ using a modified Newtonian pressure law; i.e.,

$$\begin{aligned} p_+ &= \max\{p_-,\ p_{ns}\sin^2\theta\} \\ s_+ &= s(p_{ns},\ \rho_{ns}),\ h_{ns} = h(p_{ns},\ \rho_{ns}) \\ \rho_+ &= \rho(s_+,\ p_+),\ h_+ = h(\rho_+,\ p_+) \\ V_{\sigma_+} &= -\sqrt{(V_\sigma)_{ns}^2 + 2(h_{ns} - h_+)} \end{aligned}$$

where the subscript ns indicates the values obtained from (4.10) with $\chi = 1$.

In the case $V_{\sigma-} < 0$ for a real gas, the oblique shock must be determined by iteration. The procedure used is to satisfy the energy equation

$$\tilde{u}_-^2 + 2h(\rho_-, p_-) = \tilde{u}_+^2 + 2h(\rho_+, p_+)$$

where

$$\tilde{u}_- = \sin\beta\, V_{\sigma-}\ , \quad \tilde{u}_+ = \cos\beta\, V_{\sigma-}\, (1 - \tan\theta\, \mathrm{ctn}\beta)/(\tan\theta + \mathrm{ctn}\beta)$$

$$\rho_+ = \rho_-\, \tilde{u}_-/\tilde{u}_+\ , \quad p_+ = \rho_-\, \tilde{u}_-\, (\tilde{u}_- - \tilde{u}_+) + p_-$$

and β is the shock angle (to be determined). Here $\tilde{u}_-$ and $\tilde{u}_+$ are the velocity components normal to the shock. The algorithm is initiated by taking $\sin\beta > \max\{\sin\theta, 1/m\}$; the value of $\sin\beta$ is then increased until the above equation is satisfied (if an increasing entropy solution is available). If such a solution does not exist the heuristic procedure described above for the perfect gas case is used with $\gamma = \Gamma_-$.

As we have already pointed out, the calculation downstream of body slope discontinuities is adversely affected by the resulting discontinuities in the surface flow variables. In order to minimize, these effects certain optional numerical procedures should <u>not</u> be used downstream of the discontinuity. These are:

i.) the second order accuracy option for the wall point calculation (see, sec. 3.4)

ii.) the wall entropy reduction option (see, sec. 4.2) when the discontinuity produces an oblique shock wave

Furthermore, we have found from computational experience that the version of the body surface point calculation using (3.16b) and (3.17b) gives the best results downstream of a discontinuity. Therefore, after a discontinuity in body slope has been encountered at say, Y_m, the code automatically makes the above modifications for the remainder of the calculation on $Y = Y_m$.

When discontinuities are encountered which produce large expansion discontinuities at the surface, the calculation for interior points downstream of the discontinuity becomes ill-behaved. Numerical oscillations appear in the conservation vector U which can produce negative pressure at isolated points in the interior flow field. For such situations, the code has incorporated a selective (local) smoothing scheme*. If the pressure in either the predictor or corrector step is negative at an interior point (X_n, Y_m, Z^k) then the conservation vector is redefined using

$$\tilde{U}^k_{n,m} = (U^k_{n+1,m} + \Theta\, U^k_{n,m} + U^k_{n-1,m})/(\Theta + 2) \qquad (4.10.1)$$

where Θ is a non-negative integer which is chosen by the user. In the calculation, $U^k_{n,m}$ is replaced by $\tilde{U}^k_{n,m}$ and the flow variables at (X_n, Y_m, Z^k) are redefined accordingly. In some instances, the computed results are improved after large expansion discontinuities by reducing the step size. The code includes an option which automatically does this after an expansion discontinuity

*The application of such smoothing techniques can be shown to yield first order accurate approximations for smooth flow problems.

by changing the step size to $\Delta Z/I$ where I is an integer greater than or equal to one which can be chosen by the user. The calculation for interior points downstream of compression discontinuities is better behaved than when expansion discontinuities are present and no special techniques need be applied at interior points in the case of compressions.

Immediately downstream of large compression or expansion discontinuities the wall point calculation is ill-behaved. We have found in computations for planar compression and, expansion ramps and axially-symmetric bodies that the region of poor surface results extends from the discontinuity downstream to a point on the body where the shock wave propagates into the interior flow past the first interior point (corresponding to n = 2). Based on these observations, the difficulty appears to be that in the wall point calculation the differences in the X direction must be taken across the flow discontinuity until it propagates across the first interior point. At the present time no rational approach for computing the wall points in this region which is valid for general three dimensional flows is known. We have found that the following heuristic procedure improves the numerical results at the wall in this region. After a discontinuity in body slope has been encountered, which produces a pressure discontinuity at say, Y_m, the wall point calculation at Y_m for subsequent steps is performed while the terms in (3.16) which contain X-derivatives are multiplied by a factor. This factor increases smoothly from zero at the edge to unity after a fixed number of downstream marching steps.

The user can select the number of marching steps for the code to use this procedure. (When this number is put equal to zero the procedure is not used.)

In the calculation of complex body shapes (e.g., bodies with slices and finite span flaps), there can be planes Z = constant where some wall points have expansion discontinuities, some have compression discontinuities, and some have no discontinuities. The code logic has been written to account for such situations with regard to both the determination of the discontinuities at the body surface and the special numerical procedures used downstream of the discontinuities.

4.2 Wall Entropy Reduction

In the calculation of blunted smooth body shapes, the basic scheme will maintain the wall entropy at the stagnation point value (see, (3.13), (3.14) and (3.18)). This is the correct value of wall entropy corresponding to inviscid flow. In the case of a slender blunted body, the flow variables downstream of the nose develop large radial gradients at the wall (except for pressure and the normal component of velocity). The region containing these gradients decreases in thickness as the flow develops downstream. This inviscid phenomenon is known as the entropy (or vortical) layer. As we have previously pointed out (in section 3.4) the basic computational scheme is inherently capable of maintaining the correct inviscid value of wall entropy and performing the calculation when strong entropy layers are present without using special numerical techniques. In practical calculations of interest, the entropy layer cannot actually be resolved

numerically because ultimately the layer's thickness will become smaller than the radial mesh spacing that one can afford to employ on any computer. When this happens, the present code computes a flow discontinuity at the body surface which corresponds to the variation of the flow variables across the entropy layer. This numerical discontinuity has the character of a contact discontinuity in that the pressure and normal component of velocity are continuous across it.

The development of the entropy layer and the corresponding numerical discontinuity produces at the body surface a flow which has a higher entropy and lower Mach number than the adjacent interior flow. This has at least two important effects on the subsequent calculation for the downstream flow field. First, the low axial Mach number at the surface causes a decrease in the step size ΔZ (see, (3.25). Thus, more computational steps are required and, hence, more computational time is required than if the wall entropy were at a level corresponding to the adjacent interior flow. Second, the lower speed flow on the wall can, in the presence of internal shock waves near the wall, become locally subsonic in the axial direction even though the adjacent interior flow remains supersonic. The appearance of axially subsonic flow at any point in the computation immediately halts the calculation.

It is therefore expedient in some cases to reduce the wall entropy and thereby reduce or eliminate the entropy layer. The procedure for performing this given by Kyriss and Harris (ref. 6) is incorporated as an option in the present code. The method is basically to define the wall entropy by linear extrapolation from the interior points rather than using (3.13), (3.14)

and (3.18). This procedure gradually reduces the wall entropy as the entropy layer develops. The actual algorithm is given by

$$S^{*}_{1,m} = \begin{cases} 2\,S^{k}_{2,m} - S^{k}_{3,m} \; ; & \text{if } S^{k}_{2,m} \geq S^{k}_{3,m} \\ \frac{1}{2}(S^{k}_{2,m} + S^{k}_{3,m}) \; ; & \text{if } S^{k}_{2,m} < S^{k}_{3,m} \end{cases} \tag{4.11}$$

$$S^{k+1}_{1,m} = \begin{cases} 2\,S^{*}_{2,m} - S^{*}_{3,m} \; ; & \text{if } S^{*}_{2,m} \geq S^{*}_{3,m} \\ \frac{1}{2}(S^{*}_{2,m} + S^{*}_{3,m}) \; ; & \text{if } S^{*}_{2,m} < S^{*}_{3,m} \end{cases} \tag{4.12}$$

The wall entropy is defined by (4.11) or (4.12) for $1 \leq m \leq ms$ and by (3.13) or (3.14) and (3.18) for $ms < m \leq M$ where ms is an integer $0 \leq ms \leq M$ which is selected by the user. Note that when ms = 0, the option is not used and when ms = M, the option is used for all wall points. Kyriss and Harris use essentially this procedure only on the windward plane (ms = 1, for the symmetric problem) and allow the lower entropy to be convected around the body using (3.18). We have found that ms = 2 works better in the present code. Note that unless ms = M, there will be no reduction of the wall entropy on the leeside in a symmetric problem (c.f., (3.18)).

Reducing the wall entropy by the above procedure substantially increases the step size and thus decreases computer run time without changing the surface pressure results for smooth body geometries. The other surface variables, however, are changed considerably. When discontinuities in body slope are encountered, the associated jumps in the surface flow variables (see, sec. 4.1) are significantly different depending on whether or not the wall entropy has been reduced. Further, differences in the computed

surface pressure distribution develop for a region immediately downstream of the body slope discontinuity even though the option is turned off after a discontinuity in body slope is encountered. We have also found that in some calculations involving windward flaps that the use of this option resulted in successful calculations, whereas, the calculation not using the option failed due to the appearance of subsonic axial flow at some point on the flap.

It should be pointed out here that the above option is a purely heuristic procedure and there are many open questions associated with its use. No rational interpretation of the numerical results obtained using the option has been given--the results do not correspond to inviscid flow in the strict sense. At the present time it is not known a-priori when (or when not) to use this option in general circumstances. When new flow configurations are to be computed, we strongly recommend that where possible a few test cases should be computed both with and without the option and that the decisions on the use of the option should be based on a close examination of the results for physical consistency.

4.3 Mesh Clustering

In this section the use of the mesh clustering transformations in our code is discussed. As indicated in Section 3.1, clustering the mesh in the physical plane in either r or ϕ, or both directions, can be accomplished by appropriate choices of the mappings f(X,Y,Z) and g(Y,Z). The primary purpose of mesh clustering is to distribute points in the physical plane (r,ϕ),Z = const. so that a region of large flow field gradients will contain relatively more computational points than a region where there are more gradual changes. The use of such techniques is fairly common, see refs. 3,7, but very rarely is it carefully described in terms of its overall efficiency and cost effectiveness.

Our preliminary experimentation with these techniques has revealed that a straightforward attempt at incorporating such mappings into flow field calculations may produce unexpected difficulties. For example, attempts at clustering mesh points in the r direction near the body surface to resolve the entropy layer were not successful. It is clear that if one attempts to fully resolve the types of entropy layers encountered on missile configurations that the transformations themselves would have to incorporate such large gradients that computationally the transformations could appear to be "discontinuous".

To illustrate some of the code requirements that must be satisfied an example is included. Consider the set of transformation functions for (3.2) given by

$$\bar{z} = Z$$

$$\bar{x} = f(X,Y,Z) = \frac{(Z - z_o)X^j + X}{(Z - z_o) + 1}, \quad Z \geq z_o$$

$$\bar{y} = g(Y,Z) = \frac{(Z - z_o)Y^\ell + Y}{(Z - z_o) + 1}, \quad Z \geq z_o$$

where j,ℓ are integer exponents chosen suitably and z_o represents the Z location where the calculation is first started. The above transformation has the effect that a uniform mesh at $Z = z_o$ is gradually, as Z increases, evolved into a mesh with more radial points clustered about the body surface and more planes clustered about the wind plane ($\phi = 0$) in the physical plane (see Fig. 6).

For $f(X,Y,Z)$ and $g(Y,Z)$ to be an admissible coordinate transformation they must be one-to-one (invertible) mappings and twice continuously differentiable with respect to X,Y,Z and satisfy

$$\begin{aligned} f(0,Y,Z) &= g(0,Z) = 0 \\ f(1,Y,Z) &= g(1,Z) = 1 \end{aligned} \tag{4.13}$$

For our particular example, the definition immediately reveals that these conditions are all satisfied. Note in addition to the piecewise analytical definition of f and g our code requires the following partial derivatives be similarly given:

$$\{f_X, f_Y, f_Z, f_{XX}, f_{YY}, f_{ZX}\}$$
$$\{g_Y, g_Z, g_{YY}, g_{ZY}\}$$

Returning to our above example, note that these partial derivatives are easily determined. It is also easy to verify that our example is one-to-one since one easily checks that $f_X(X,Y,Z) > 0$ and $g_Y(Y,Z) > 0$ are satisfied. Furthermore, we observe that

$$f_X(0,Y,Z) = g_Y(0,Z) = \frac{1}{Z+1}$$

indicating that for $Z >> 1$ more points are clustered about th ly surface and wind plane (see Fig. 6).

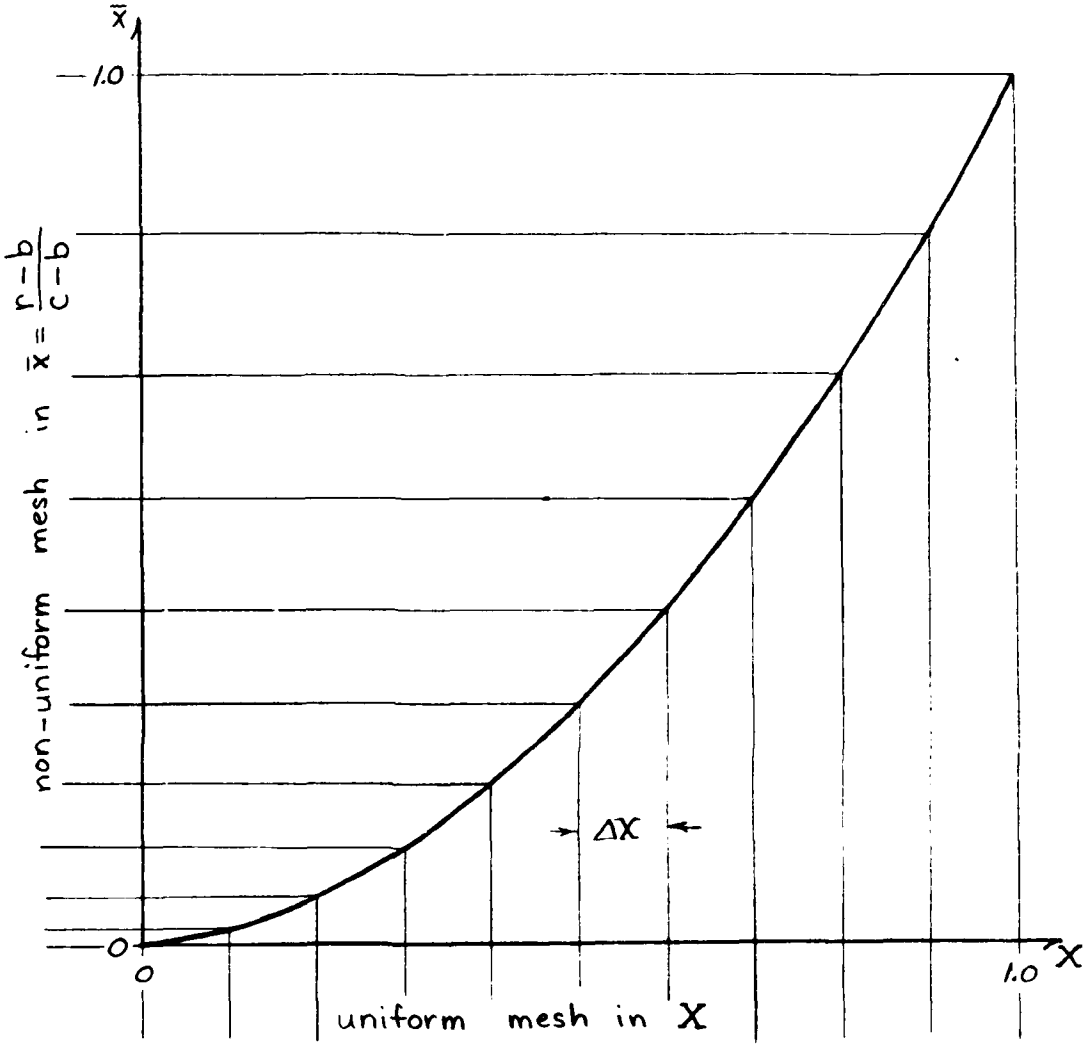


Fig. 6. The clustering function f given in (4.13) for j = 2, z_o = 1, and Z = 10.

Finally an additional condition is imposed at the boundary. To apply the transformation to a symmetric problem it is required that

$$f_Y(X,0,Z) = f_Y(X,1,Z) = 0$$

For the above example this is clearly satisfied. On the other hand, the above transformation would not be suitable for a non-symmetric problem. For in this case it is required that

$$\begin{aligned} f(X,Y+1,Z) &= f(X,Y,Z) \\ g(Y,Z) - Y &= g(Y+1,Z) - (Y+1) \ . \end{aligned} \tag{4.14}$$

Whereas the former condition is clearly satisfied, it is easy to see that no choice of exponents ℓ in our above example will make $g(Y,Z) - Y$ satisfy the periodicity condition.

In our early studies we attempted to systematically develop transformations from simple piecewise polynomial or rational functions. The use of polynomials would result in a saving on the functional evaluations required compared to using transcendental mapping functions (refs. 3,7). Our experiences with implementing a variety of polynomial interpolatory methods revealed that it was difficult to predict the parameters for a particular computation. Moreover, with the various analytical conditions to be satisfied (such as (4.14)) and the further desirability of having specific points included in the mesh, the analytical approach becomes overly burdened. For these, and other reasons stated below, we have adopted the approach of defining mesh transformations by a user-given set of discrete spatial points. The user must provide a set of points $\{\bar{x}_n\}$ and/or $\{\phi_m = \bar{y}_m \ \phi_0\}$ and the code (see sections 12.5, 12.6) will assign the relationship

$$\bar{x}_n = f(X_n), \ n = 1,2\cdots N; \ \bar{y}_m = g(X_m), \ m = 1,2\cdots M.$$

Clearly one must select the points to satisfy (4,13) and whatever appropriate conditions such (4.14) may be required.

Note well, this implicit definition of the underlying transformation functions will be assumed to be smooth. If the data chosen does not arise from a context where smoothness is guaranteed, it is best to modify the choice to enforce some degree of smoothness. Our strategy has been to implement a "mesh pre-processor". All this entailed was a simple smoothing of the data points using equation (4.10.1) with $\Theta = 2$. This is repeated several times until the resulting differences appear smooth.

To approximate derivatives we use standard finite difference approximations from the given (or smoothed) data. For our purposes it is sufficient to approximate these derivatives to second order accuracy. For interior points $n = 2,\cdots,N-1$, $m = 2,\cdots,M-1$ we use standard centered differences

$$(f_X)_n \sim \frac{f(X_{n+1}) - f(X_{n-1})}{2\Delta X}$$

$$(f_{XX})_n = \frac{f(X_{n+1}) - 2f(X_n) + f(X_{n-1})}{\Delta X^2}$$

with similar expressions for $(g_Y)_m$ and $(g_{YY})_m$.

The only points requiring any explanation are the end points of the intervals. By second order extrapolation

$$\begin{aligned}(f_{XX})_{n=1} &= 2(f_{XX})_{n=2} - (f_{XX})_{n=3} \\ (f_{XX})_{n=N} &= 2(f_{XX})_{n=N-1} - (f_{XX})_{n=N-2}\end{aligned} \tag{4.15}$$

A second order approximation to $(f_X)_{i=0}$ can be derived by assuming that (f_{XX}) is essentially linear on the intervals $(0, 2\Delta X)$ and $(1 - 2\Delta X, 1)$ then

$$(f_X)_{n=1} = (f_X)_{n=2} - \frac{\Delta X}{2} ((f_{XX})_{n=1} + (f_{XX})_{n=2}) \tag{4.16a}$$

and

$$(f_X)_{n=N} = (f_X)_{n=N-1} - \frac{\Delta X}{2} ((f_{XX})_{n=N} + (f_{XX})_{n=N-1}) \tag{4.16b}$$

For g(Y) a similar treatment is employed, for the symmetric case. Indeed at m = 1,M the approximations identical to (4.15) and (4.16) are made to g_{YY} and g_Y, respectively. In the symmetric case it is further required to compute fringe planes $Y = -\Delta Y$ and $Y = 1 + \Delta Y$. Again the same approximations are used except now we assume g_{YY} is linear in the extended interval $(-\Delta Y, 2\Delta Y)$ and $(1-2\Delta Y, 1+\Delta Y)$. In the non-symmetric case, the data defining g(Y) is extended beyond $0 \leq Y \leq 1$ using the perodicity conditions. The first and second derivatives at Y = 0,1 are determined in the same manner as at interior points.

Our experience has convinced us that the use of the above type of discrete definition of mesh transformations in conjunction with a pre-processor mesh smoother offers the widest flexibility for the user. One can easily create a local refinement of the mesh without much analytical care if one is prepared to allow a mesh smoother to alter your initial mesh array. Our experience has shown that a reasonable eyeball choice followed by several smoothings will only slightly alter the mesh and will retain the concentrations in the regions where originally desired.

5. FORCE AND MOMENT CALCULATIONS

The aerodynamic forces and moments acting on the vehicle are obtained by numerically integrating the computed surface pressure distributions. In the code, these calculations are performed in auxiliary subroutines so that the user may conveniently substitute alternative definitions and/or numerical integration procedures. In this section, the procedures and definitions currently available in the code are described.

The sign conventions for the components of the aerodynamic force and moment are illustrated in Fig. 7. The force and moment are computed assuming that the base pressure is p_∞. Both the force and moment have dimensions of pressure. The center for the moments, C, is any point on the z-axis where, say, $z=z_c$ (see, Fig. 7). The derivative with respect to z of the force components are given by

$$\frac{\partial F_a}{\partial z} = \int_0^{2\pi} (p_w - p_\infty) b b_z \, d\phi \,, \tag{5.1}$$

$$\frac{\partial F_n}{\partial z} = \int_0^{2\pi} (p_w - p_\infty)(b \cos\phi + b_\phi \sin\phi) \, d\phi \,, \tag{5.2}$$

$$\frac{\partial F_y}{\partial z} = \int_0^{2\pi} (p_w - p_\infty)(b_\phi \cos\phi - b \sin\phi) \, d\phi. \tag{5.3}$$

The derivatives with respect to z of the moment components taken about $C(z=z_c)$ are given by

$$\frac{\partial M_a^c}{\partial z} = - \int_0^{2\pi} (p_w - p_\infty) b b_\phi \, d\phi \,, \tag{5.4}$$

$$\frac{\partial M_n^c}{\partial z} = (z_c - z)\frac{\partial F_y}{\partial z} + \int_0^{2\pi} (p_w - p_\infty) b^2 b_z \sin\phi \, d\phi \tag{5.5}$$

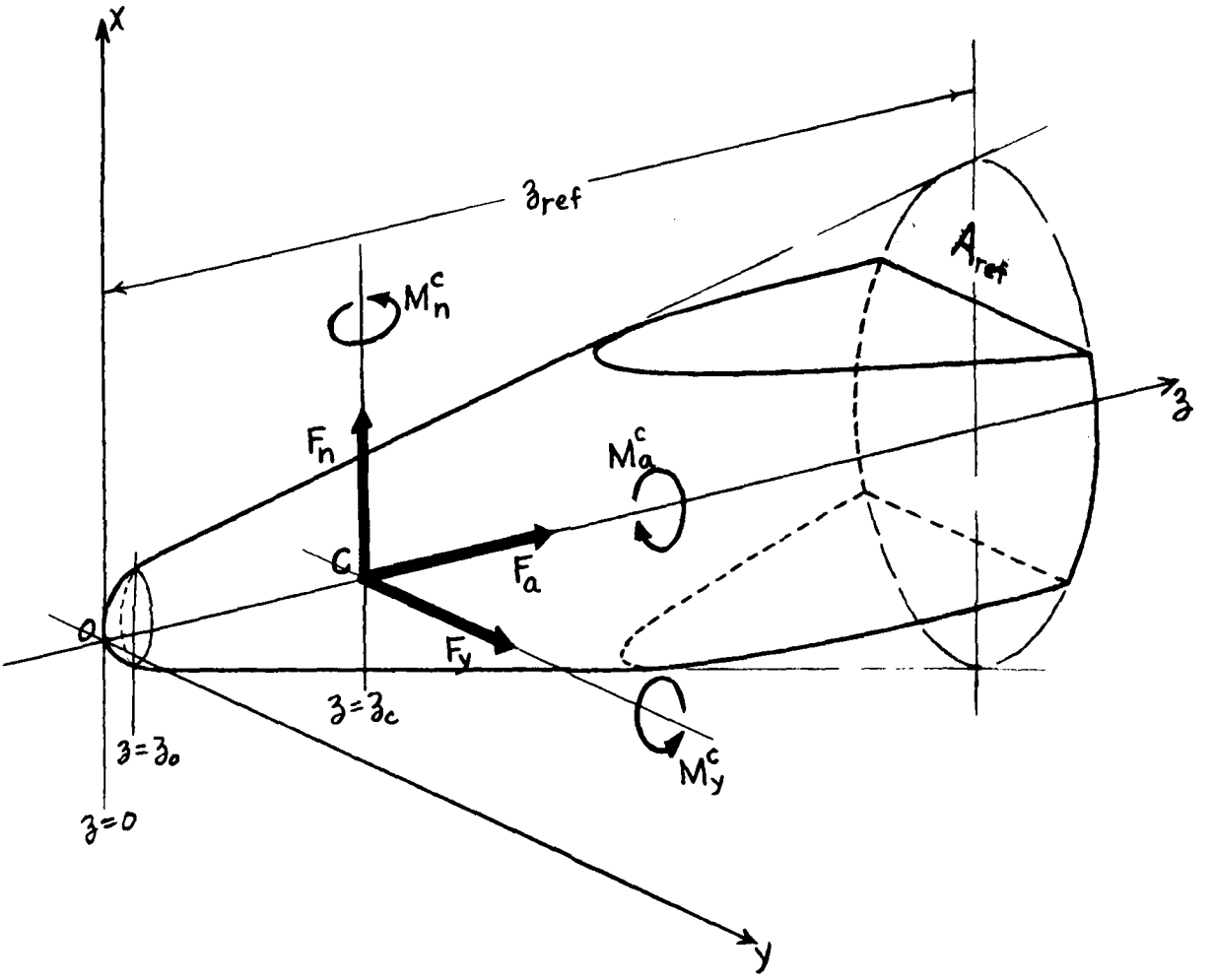


Fig. 7. Definitions of aerodynamic force and moment components

$$\frac{\partial M_y^c}{\partial z} = (z-z_c)\frac{\partial F_n}{\partial z} + \int_0^{2\pi}(p_w - p_\infty)b^2 b_z \cos\phi \, d\phi \qquad (5.6)$$

In (5.1) - (5.6), p_w denotes the surface pressure. Note that for the symmetric problem

$$\frac{\partial F_y}{\partial z} = \frac{\partial M_a^c}{\partial z} = \frac{\partial M_n^c}{\partial z} = 0 \ .$$

The integrals appearing in the above are computed numerically at each computational step, $z=Z^k$, using Simpson's rule in the computational plane. For the purpose of discussion, consider

$$\Psi = \int_0^{2\pi}\psi(\phi) \, d\phi$$

which is, for a fixed value of z, the general form of all the above integrals. For the symmetric problem, the non-zero integrals are written in the computational plane as

$$\Psi = 2\int_0^1(\psi/Y_\phi)dY = 2\int_0^1\tilde{\psi}(Y)dY \ .$$

This is numerically integrated on the uniform computational mesh using Simpson's rule in the form

$$\Psi = \frac{2\Delta Y}{3}[\tilde{\psi}_1 + 4\tilde{\psi}_2 + 2\tilde{\psi}_3 + 4\tilde{\psi}_4 + 2\tilde{\psi}_5 + \cdots$$

$$\cdots + 2\tilde{\psi}_{M-2} + 4\tilde{\psi}_{M-1} + \tilde{\psi}_M], \text{ if M is odd}$$

and

$$\Psi = \frac{2\Delta Y}{3}[\tilde{\psi}_1 + 4\tilde{\psi}_2 + 2\tilde{\psi}_3 + 4\tilde{\psi}_4 + 2\tilde{\psi}_5 + \cdots$$

$$\cdots + 2\tilde{\psi}_{M-3} + 4\tilde{\psi}_{M-2} + \frac{1}{2}(5\tilde{\psi}_{M-1} + 3\tilde{\psi}_M)], \text{ if M is even}$$

where $\tilde{\psi}_m = \tilde{\psi}(Y_m)$, $Y_M = 1$. In the last expression, the trapezoidal rule is used for the subinterval $[Y_{M-1}, Y_M]$. In the nonsymmetric problem Ψ is written as

$$\Psi = \int_0^1 (\psi/Y_\phi)dY = \int_0^1 \tilde{\psi}(Y)dY \ .$$

In this case, the integrands are periodic functions of Y with period 1 (i.e., $\tilde{\psi}(Y_1) = \tilde{\psi}(Y_M)$) and Simpson's rule becomes

$$\Psi = \frac{\Delta Y}{3}[2\tilde{\psi}_1 + 4\tilde{\psi}_2 + 2\tilde{\psi}_3 + 4\tilde{\psi}_4 + \cdots$$

$$\cdots + 2\tilde{\psi}_{M-2} + 4\tilde{\psi}_{M-1}], \text{ if M is odd}$$

and

$$\Psi = \frac{\Delta Y}{3}[6\tilde{\psi}_1 + 3\tilde{\psi}_2 + 2\tilde{\psi}_3 + 4\tilde{\psi}_4 + 2\tilde{\psi}_5 + \cdots$$

$$\cdots + 2\tilde{\psi}_{M-3} + 4\tilde{\psi}_{M-2}] \quad , \text{ if M is even.}$$

The code also computes, at each computational step, Z^k, the force and moment vectors acting on the body truncated at $z = Z^k$. These quantities are defined, for example, by

$$F_a(Z^k) = \int_0^{Z^k} \frac{\partial F_a}{\partial z}\, dz \tag{5.7}$$

with similar expressions for the other truncated force and moment components. The integrals of the type (5.7) are evaluated numerically using the trapezoidal rule; i.e.,

$$F_a(Z^{k+1}) = F_a(Z^k) + \left(\frac{Z^{k+1} - Z^k}{2}\right)\left[\left(\frac{\partial F_a}{\partial z}\right)_{z=Z^k} + \left(\frac{\partial F_a}{\partial Z}\right)_{z=Z^{k+1}}\right] \tag{5.8}$$

with similar expressions for the other force and moment coefficients. Note that this calculation requires the force and moment on the body truncated at the initial plane $z = z_o$. These quantities must be given along with the initial flow field data.

The final results are presented in coefficient form by dividing the force components and their derivatives by $\frac{\rho_\infty}{2}V_\infty^2 A_{ref}$ and the moment components and their derivatives by $\frac{\rho_\infty}{2}V_\infty^2(A_{ref})(z_{ref})$ where A_{ref} is a reference area

(non-dimensionalized by R_o^2). The parameters z_c, z_{ref}, and A_{ref} can be selected by the user; however, if no selection is made, the code assumes that $z_c = 0$, z_{ref} is the body length, and A_{ref} is the base area of the baseline body (see, Fig. 6). The code also computes two centers of pressure. The center of pressure in pitch defined by

$$(Z_{c.p.})_p = [z_c + M_y^c/F_n] \tag{5.9}$$

for $F_n \neq 0$ and the center of pressure in yaw defined by

$$(Z_{c.p.})_y = [z_c - M_n^c/F_y] \tag{5.10}$$

for $F_y \neq 0$.

APPENDIX A

DIFFERENTIAL EQUATIONS FOR BOUNDARY POINTS ON THE BODY SURFACE AND THE BOW SHOCK WAVE

In this appendix differential equations are derived which are used for the computation at points on the boundaries X = 0 and 1. These equations are obtained from certain characteristic compatability conditions evaluated on X = 0 and 1. Such equations have been used in various forms in many non-conservation codes (e.g., see refs. (6, 8, 9). The equations used in the present code differ from those previously used. The advantages of the present equations are discussed in Secs. 3.3 and 3.4.

Since we are concerned with the boundary conditions on the planes X = 0 and 1, it is sufficient to consider the system (3.3) in the form

$$\frac{\partial U}{\partial Z} + \frac{\partial F}{\partial X} = -\frac{\partial G}{\partial Y} - E \equiv \mathcal{R} . \qquad \text{(A-1)}$$

In this equation, the right hand side, $\mathcal{R}$, will be treated as an inhomogeneous term since the Y derivative, being interior to X = 0 and 1, depends only on quantities on the boundary planes. From this point of view (A-1) is treated as a hyperbolic system in one space dimension X with Z time-like. We have on the boundary plane, X = 0, the condition that $u = b_z w + (b_\phi/b)v$. On the boundary X = 1, the Rankine-Hugoniot conditions (2.7) are satisfied. To motivate our treatment of the boundary points, consider as a prototype problem the mixed initial-boundary value problem on the strip $Z \geq 0$, $0 \leq X \leq 1$ where initial data is given on Z = 0 and boundary conditions are given on X = 0 and 1. It is known (see, for example, ref. 10, pp. 471-475) from the theory of hyperbolic systems in one space dimension that, for a well posed mixed initial-boundary value problem, of this form, certain information concerning the solution at boundary points, say, (0,Z*) and (1,Z*) can be determined from the characteristic compatability conditions

corresponding to the characteristics through (0,Z*) and (1,Z*) which enter the region $\{0 \leq X \leq 1,\ Z < Z^*\}$ (e.g., in Fig. A-1, the curves $\mathcal{C}_1$ and $\mathcal{C}_3$ through (0,Z*) and the curves $\mathcal{C}_2$ and $\mathcal{C}_3$ through (1,Z*)). The remaining information required to completely determine all the unknowns at (0,Z*) and (1,Z*) must be provided by the boundary conditions specified on X = 0 and X = 1. These boundary conditions are needed to replace the characteristics through (0,Z*) and (1,Z*) which do not enter the region $\{0 \leq X \leq 1,\ Z < Z^*\}$, (e.g., $\mathcal{C}_2$ through (0,Z*) and $\mathcal{C}_1$ through (1,Z*) in Figure A-1). These characteristics are inadmissible since there is no information available for X < 0 and X > 1.

As we shall see in the following, at the boundaries X = 0 and 1 the system (A-1) has four independent characteristic compatability equations. At X = 0, there is one $\mathcal{C}_1$-type characteristic, one $\mathcal{C}_2$-type characteristic, and two $\mathcal{C}_3$-type characteristics (see Fig. A-1). Consistent with the above remarks, we will find that the three compatability conditions corresponding to the $\mathcal{C}_1$ and $\mathcal{C}_3$ characteristics will provide differential equations for advancing three unknown quantities on X = 0. These quantities together with the boundary condition $u = b_z w + (b_\phi/b)v$ determine all flow variables on X = 0.

On the boundary X = 1, there are three $\mathcal{C}_1$-type characteristics and one $\mathcal{C}_2$-type characteristic (there are no $\mathcal{C}_3$-types). We will find that the compatability condition corresponding to the $\mathcal{C}_2$-type characteristic will provide a differential equation for determining the shock shape function $c(\phi,z)$. The shock geometry together with the Rankine-Hugoniot relations (2.7) determine all flow quantities on X = 1.

In order to derive the necessary characteristic conditions, the system (A-1) is rewritten in an equivalent quasi-linear form. This can be done by changing the dependent variables in (A-1) from U to

$$Q = \begin{pmatrix} p \\ u \\ v \\ w \end{pmatrix} .$$

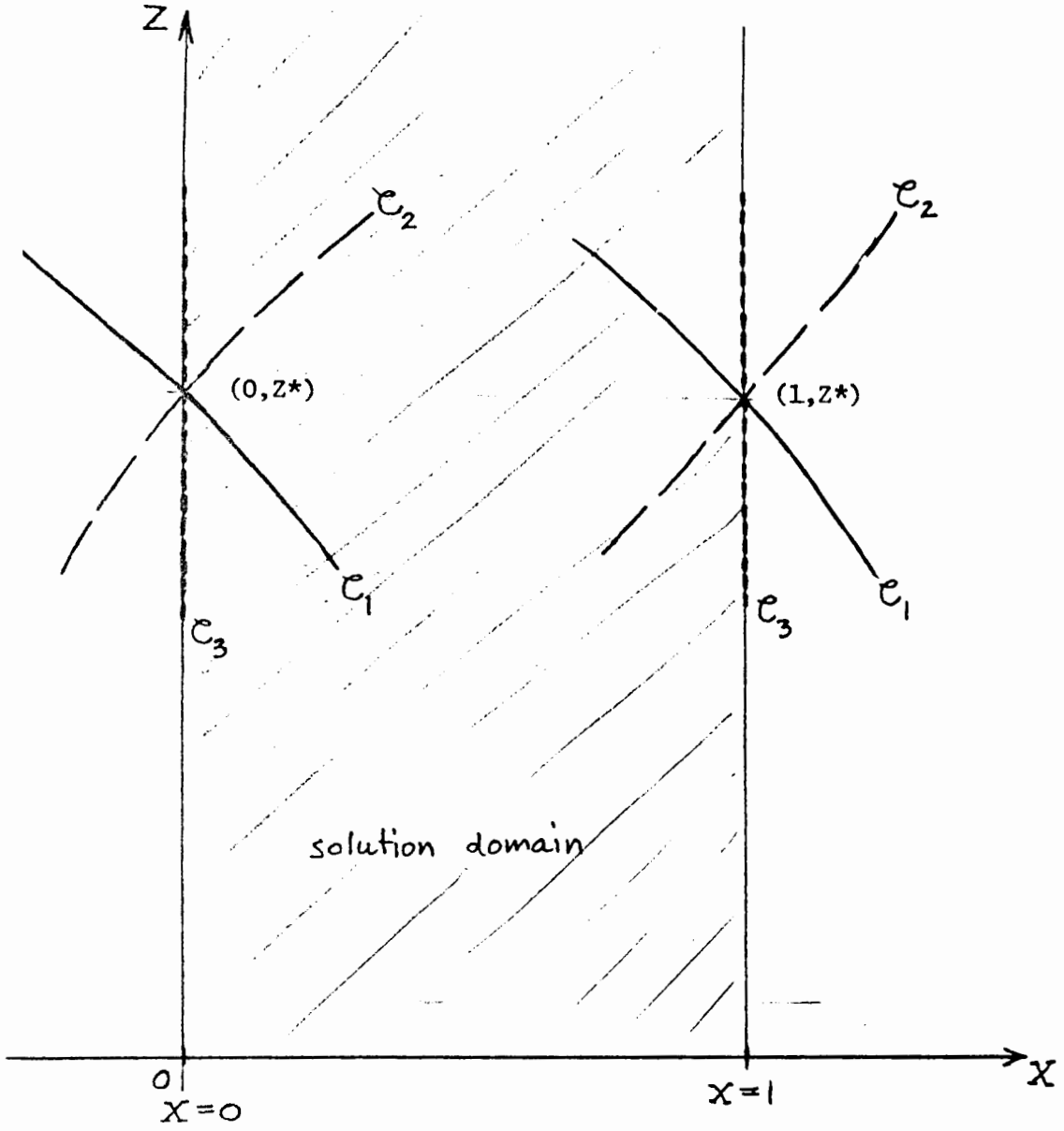


Fig. A-1. Types of characteristic curves through boundary points on X = 0 and 1

We assume here that $\rho = \rho(p,h)$ where h is given by (2.2). Using the chain rule, we have

$$\frac{\partial U}{\partial Z} = \left[\frac{\partial U}{\partial Q}\right] \frac{\partial Q}{\partial Z} \tag{A-2}$$

and, from (3.3a),

$$\frac{\partial F}{\partial X} = \left(X_z \left[\frac{\partial U}{\partial Q}\right] + X_r \left[\frac{\partial \mathcal{F}}{\partial Q}\right] + X_\phi \left[\frac{\partial \mathcal{G}}{\partial Q}\right]\right) \frac{\partial Q}{\partial X}$$

$$+ \frac{\partial X_z}{\partial X} U + \frac{\partial X_r}{\partial X} \mathcal{F} + \frac{\partial X_\phi}{\partial X} \mathcal{G} - \frac{1}{r} X_\phi \frac{\partial r}{\partial X} \mathcal{G} \tag{A-3}$$

where $\left[\frac{\partial U}{\partial Q}\right]$, $\left[\frac{\partial \mathcal{F}}{\partial Q}\right]$, and $\left[\frac{\partial \mathcal{G}}{\partial Q}\right]$ are Jacobian matrices.

These can be obtained by taking partial derivatives with respect to the components of Q of (2.1a) and (2.1b) using (2.2); for example,

$$\left[\frac{\partial U}{\partial Q}\right] = \begin{bmatrix} \mathcal{K}_2 w & -\mathcal{K}_1 wu & -\mathcal{K}_1 wv & \rho - \mathcal{K}_1 w^2 \\ 1+\mathcal{K}_2 w^2 & -\mathcal{K}_1 w^2 u & -\mathcal{K}_1 w^2 v & 2\rho w - \mathcal{K}_1 w^3 \\ \mathcal{K}_2 uw & \rho w - \mathcal{K}_1 wu^2 & -\mathcal{K}_1 wuv & \rho u - \mathcal{K}_1 w^2 u \\ \mathcal{K}_2 wv & -\mathcal{K}_1 wvu & \rho w - \mathcal{K}_1 wv^2 & \rho v - \mathcal{K}_1 w^2 v \end{bmatrix}$$

where $\mathcal{K}_1 = \left(\frac{\partial \rho}{\partial h}\right)_p$ and $\mathcal{K}_2 = \left(\frac{\partial \rho}{\partial p}\right)_h$ are thermodynamic quantities which are related to the sound speed by

$$\frac{\rho}{a^2} = \rho \mathcal{K}_2 + \mathcal{K}_1 \ . \tag{A-4}$$

Substitution of (A-2) and (A-3) into (A-1) yields the desired quasi-linear system. This system is simplified by multiplying each term on the left by the non-singular matrix

$$\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -w & 1 & 0 & 0 \\ -u & 0 & 1 & 0 \\ -v & 0 & 0 & 1 \end{bmatrix}$$

The resulting system of equations is

$$\mathbf{A}\frac{\partial Q}{\partial Z} + \mathbf{B}\frac{\partial Q}{\partial X} = \tilde{\mathbf{R}} \tag{A-5}$$

where

$$\mathbf{A} = \begin{bmatrix} \kappa_2 w & -\kappa_1 wu & -\kappa_1 wv & \rho - \kappa_1 w^2 \\ 1 & 0 & 0 & \rho w \\ 0 & \rho w & 0 & 0 \\ 0 & 0 & \rho w & 0 \end{bmatrix} \tag{A-5a}$$

$$\mathbf{B} = \begin{bmatrix} \kappa_2 A & X_r \rho - u\kappa_1 A & \frac{1}{r} X_\phi \rho - v\kappa_1 A & X_z \rho - w\kappa_1 A \\ X_z & 0 & 0 & \rho A \\ X_r & \rho A & 0 & 0 \\ \frac{1}{r} X_\phi & 0 & \rho A & 0 \end{bmatrix} \tag{A-5b}$$

and

$$\tilde{\mathbf{R}} = -\mathbf{D}\frac{\partial G}{\partial Y} - \mathbf{D}\mathcal{E} + \frac{\partial Y_z}{\partial Y}\mathbf{D}U + \left(\frac{\partial Y_\phi}{\partial Y} + \frac{1}{r} X_\phi \frac{\partial r}{\partial X}\right)\mathbf{D}\mathcal{U} \tag{A-5c}$$

The system (A-5) can be reduced to characteristic form by standard methods (see, e.g., ref. 10 , pp. 424-427) although the required computations are lengthy. We summarize the procedure here. Since the matrix is non-singular for w>a, we can consider the characteristic matrix $\mathbf{A}^*$ associated with (A-5) in the form $\mathbf{A}^* = \lambda\mathbf{A} + \mathbf{B}$ where λ is a scalar. For every real value of λ which is a root of

$$\det(\mathbb{A}^*) = \rho^3(\lambda w + A)^2\{(\lambda w + A)^2/a^2 - [(\lambda + X_z)^2 + X_r^2 + \frac{1}{r^2} X_\phi^2]\} = 0 \qquad \text{(A-6)}$$

there corresponds a real left null vector, ℓ_λ, of $\mathbb{A}^*$ (i.e., $\ell_\lambda \mathbb{A}^* = 0$). Hence,

$$\ell_\lambda \mathbb{B} = - \lambda \ell_\lambda \mathbb{A} \ . \qquad \text{(A-7)}$$

Multiplying (A-5) on the left by ℓ_λ and using (A-7), we obtain the scalar equation

$$\ell_\lambda \mathbb{A} (\frac{\partial Q}{\partial Z} - \lambda \frac{\partial Q}{\partial X}) = \ell_\lambda \tilde{R} \ . \qquad \text{(A-8)}$$

This equation is referred to as the characteristic compatability condition along the characteristic curve defined by $\frac{dX}{dZ} = -\lambda$ since the quantity $\frac{\partial Q}{\partial Z} - \lambda \frac{\partial Q}{\partial X}$ is the directional derivative tangent to this curve. The roots of (A-6) are

$$\left.\begin{aligned} &\lambda_o = - A/w \text{ (multiplicity 2)} \\ &\text{and} \\ &\lambda_\pm = \frac{-wA + a^2 X_z \pm a^2 \beta_1}{w^2 - a^2}, \ \beta_1 = \sqrt{\frac{(w^2 - a^2)(X_r^2 + \frac{1}{r^2} X_\phi^2) + (A - X_z w)^2}{a^2}} \ . \end{aligned}\right\} \qquad \text{(A-9a)}$$

Two independent null vectors correspond to the multiple root λ_o; viz.,

$$\ell_{\lambda_o}^{(1)} = (0,0,\frac{1}{r} X_\phi, - X_r), \ \ell_{\lambda_o}^{(2)} = (0,w,u,v) \ . \qquad \text{(A-9b)}$$

A left null vector corresponding to $\lambda_\pm$ is

$$\begin{aligned} \ell_{\lambda_\pm} = &((\lambda_\pm w + A)(2\rho - \mathcal{K}_1 V^2), - \rho(\lambda_\pm + X_z) + \mathcal{K}_1 w(\lambda_\pm w + A), \\ &- \rho X_r + \mathcal{K}_1 u(\lambda_\pm w + A), - \frac{\rho}{r} X_\phi + \mathcal{K}_1 v(\lambda_\pm w + A)). \end{aligned} \qquad \text{(A-9c)}$$

The standard form for the compatability equations (A-8) is to introduce the associated directional derivatives. Used in this form the boundary point computations would be essentially as in a reference plane method of characteristics. Such a procedure is complicated since it requires a considerable amount of interpolation. The method used herein eliminates this difficulty by considering the partial differential equations (A-8) directly. These equations are rewritten on the surfaces X = 0 and 1 using the boundary conditions.* The resulting equations, valid only on X = 0 and 1, are solved numerically using predictor-corrector methods similar to those used for interior points.

Wall Boundary Points (X = 0)

On the boundary X = 0, $u = b_z w + (b_\phi/b)v$ which implies that A = 0 on X = 0 (c.f., eqs. (3.3d) and (3.3g)). Thus on X = 0, $\beta_1 > |X_z|$ (since w > a) which implies that $\lambda_+ > 0$ and $\lambda_- < 0$. Also on X = 0, $\lambda_o = 0$. Note that on X = 0, λ_o corresponds to a streamline since X = 0 is a stream surface. Referring to Fig. (A-1), λ_+ corresponds to the characteristic curve $\mathcal{C}_1$, λ_- corresponds to $\mathcal{C}_2$, and λ_o corresponds to $\mathcal{C}_3$. According to our earlier remarks, we exclude from further consideration the root λ_-.

We consider first the compatability condition (A-8) with $\ell_\lambda = \ell_{\lambda_+}$. The left-hand side of (A-8) is, using (A-5a) and (A-9),

$$\rho\{\beta_1(\frac{\partial p}{\partial Z} - \lambda_+ \frac{\partial p}{\partial X}) - \rho w\ (X_z\frac{\partial w}{\partial Z} + X_r\frac{\partial u}{\partial Z} + \frac{1}{r}X_\phi\frac{\partial v}{\partial Z}) + \rho w\lambda_+(X_z\frac{\partial w}{\partial X} + X_r\frac{\partial u}{\partial X} + \frac{1}{r}X_\phi\frac{\partial v}{\partial X}) + \rho A(\frac{\partial w}{\partial Z} - \lambda_+\frac{\partial w}{\partial X})\}\ .$$

Note that at the boundary X = 0, we have using the boundary condition and (3.3g) that

$$X_z\frac{\partial w}{\partial Z} + X_r\frac{\partial u}{\partial Z} + \frac{1}{r}X_\phi\frac{\partial v}{\partial Z} = X_r(-b_z\frac{\partial w}{\partial Z} + \frac{\partial u}{\partial Z} - \frac{b\phi}{b}\frac{\partial v}{\partial Z}) = X_r\ (w\frac{\partial b_z}{\partial Z} + v\frac{\partial b_\phi/b}{\partial Z})$$

where $\frac{\partial b_z}{\partial Z}$ and $\frac{\partial b_\phi/b}{\partial Z}$ depend only on the given body geometry. Using the above, we can write the compatability condition corresponding to λ_+ at the

*This approach was originally suggested by Kentzer (ref. 11).

wall, X = 0, in the form

$$\beta_1 \left(\frac{\partial p}{\partial Z} - \lambda_+ \frac{\partial p}{\partial X}\right) + \rho w \lambda_+ \frac{\partial A}{\partial X} = \rho w \left(w \frac{\partial b_z}{\partial Z} + v \frac{\partial b_\phi / b}{\partial Z}\right) X_r$$

$$+ \rho w \lambda_+ \left[w \frac{\partial X_z}{\partial X} - u \frac{\partial X_r}{\partial X} - v \frac{\partial (X_\phi / r)}{\partial X}\right] + \frac{1}{\rho} \ell_{\lambda_+} \tilde{\mathcal{R}} \qquad \text{(A-10)}$$

The completely expanded form of (A-10) given by (3.16) is used to advance the pressure on the wall.

The left-hand side of (A-8) corresponding to $\ell_\lambda = \ell^{(1)}_{\lambda_o}$ is given by

$$\rho w \left\{\left(\frac{1}{r} X_\phi \frac{\partial u}{\partial Z} - X_r \frac{\partial v}{\partial Z}\right) - \lambda_o \left(\frac{1}{r} X_\phi \frac{\partial u}{\partial X} - X_r \frac{\partial v}{\partial X}\right)\right\}$$

On X=0, $\lambda_o = 0$ and $\frac{1}{r} X_\phi = - X_r (b_\phi / b)$. Hence, the compatibility condition corresponding to $\ell_\lambda = \ell^{(1)}_{\lambda_o}$ at the wall can be written as

$$\rho w X_r \left(\frac{\partial V_2}{\partial Z} - u \frac{\partial (b_\phi / b)}{\partial Z}\right) = \ell^{(1)}_{\lambda_o} \tilde{\mathcal{R}} \qquad \text{(A-11)}$$

where

$$V_2 = v + (b_\phi / b) u \qquad \text{(A-11a)}$$

The completely expanded form of this equation is given in (3.18). Equation (A-11) is used to advance the quantity V_2 on the wall. Note that V_2 is, except for a factor involving only body geometry, a component of velocity tangent to the wall.

The compatability equation corresponding to $\ell_\lambda = \ell^{(2)}_{\lambda o}$ can be written, by direct expansion of (A-8), as

$$w \left(\frac{\partial p}{\partial Z} + \frac{\rho}{2} \frac{\partial V^2}{\partial Z}\right) + A \left(\frac{\partial \rho}{\partial X} + \frac{\rho}{Z} \frac{\partial V^2}{\partial X}\right) = - B \left(\frac{\partial p}{\partial Y} + \frac{\rho}{2} \frac{\partial V^2}{\partial Y}\right)$$

where $V^2 = w^2 + u^2 + v^2$. The above equation involves only derivatives of thermodynamic quantities. Indeed, using (2.2) and (A-4), we have (since $\rho = \rho(p,h)$)

$$dp + \frac{\rho}{2}\, dV^2 = dp - \rho\, dh = \frac{\rho}{\mathcal{K}_1}\left(\frac{1}{a^2}\, dp - d\rho\right). \qquad \text{(A-12)}$$

Introducing the entropy, s, we have

$$-\left(\frac{\partial \rho}{\partial s}\right)_p ds = \frac{1}{a^2}\, dp - d\rho$$

which with (A-12) implies that the compatability equation can be written as

$$w \frac{\partial s}{\partial Z} + A \frac{\partial s}{\partial X} = - B \frac{\partial s}{\partial Y}\ .$$

This equation expresses, in the computational space, that the rate of change of entropy on a streamline is zero. On the boundary X=0, we have (since A=0)

$$w \frac{\partial s}{\partial Z} = - B \frac{\partial s}{\partial Y} \qquad \text{(A-13)}$$

Note that eqs. (A-10), (A-11) and (A-13) can be used to advance p, V_2, and s at the wall points. From the values of p and s, the quantities ρ and h can be determined. The magnitude of the velocity vector can then be determined using (2.2). The quantity V_2 and the boundary condition then give the velocity components. This procedure is described in detail in section 3.4.

Shock Boundary Points (X = 1)

On the boundary X = 1,

$$A = (- c_z w + u - c_\phi v/c)X_r = - V_n \nu_s X_r$$

(c.f., (3.3d), (3.3g)). Since $V_n > 0$, it follows from (4.9a) that $\lambda_o > 0$. On X = 1, the roots $\lambda_\pm$ of (A-6) are (see (A-9a))

$$\lambda_\pm = \frac{(wV_n \nu_s - a^2 c_z)X_r \pm a^2 \beta_1}{w^2 - a^2}$$

where

$$\beta_1^2 = \frac{X_r^2}{a^4}\left[(wV_n\nu_s - a^2 c_z)^2 + \nu_s^2 (w^2 - a^2)(a^2 - V_n^2)\right].$$

Since $w > a$ and shock theory gives $V_n \leq a$, it follows that $a^2\beta_1 \geq X_r (wV_n\nu_s - a^2 c_z) > 0$. Hence $\lambda_+ > 0$ and $\lambda_- \leq 0$ (equality when $V_n = a$ which corresponds to zero shock strength). Referring to Fig. (A-1), λ_o and λ_+ correspond to characteristic curves of the type $\mathcal{C}_1$, and thus, by our earlier remarks, we exclude these roots from further consideration.

The compatability condition (A-8) with $\ell_\lambda = \ell_{\lambda-}$ can be written using (A-5a), (A-7) and (A-9b) as

$$\beta_1 \frac{\partial p}{\partial Z} + \rho\left[X_r (w \frac{\partial u}{\partial Z} - u \frac{\partial w}{\partial Z}) + \frac{1}{r} X_\phi (w \frac{\partial v}{\partial Z} - v \frac{\partial w}{\partial Z})\right] = -\frac{1}{\rho} \ell_{\lambda-} (\tilde{\mathcal{R}} - \mathcal{B}) \tag{A-14}$$

Note that (A-3), (A-5a), and (3.3c) imply that

$$\tilde{\mathcal{R}} - \mathcal{B} = -(\frac{\partial F}{\partial X} + \frac{\partial G}{\partial Y} + E).$$

Consider now the left hand side of (A-14) evaluated on the surface $X = 1$. Since the Rankine-Hugoniot conditions are satisfied identically on $X = 1$, these relations can be differentiated with respect to Z in order to obtain expressions for $\frac{\partial p}{\partial Z}$, $\frac{\partial u}{\partial Z}$, $\frac{\partial v}{\partial Z}$, and $\frac{\partial w}{\partial Z}$ on $X = 1$. Substitution of these expressions into (A-14) yields a partial differential equation for the shock slopes c_z and c_ϕ. We summarize the development of this equation here. From the shock relations (3.9), we have on $X = 1$ that $w = w_\infty - (V_{n_\infty} - V_n)e_1$, $u = u_\infty + (V_{n_\infty} - V_n)e_2$, and $v = v_\infty - (V_{n_\infty} - V_n)e_3$ where $e_1 = c_z/\nu_s$, $e_2 = 1/\nu_s$, and $e_3 = c_\phi/(c\nu_s)$. Differentiating these with respect to Z, we obtain that on $X = 1$

$$\left.\begin{aligned}
\frac{\partial w}{\partial Z} &= -\left(\frac{\partial V_{n_\infty}}{\partial Z} - \frac{\partial V_n}{\partial Z}\right) e_1 - (V_{n_\infty} - V_n)\frac{\partial e_1}{\partial Z} \\
\frac{\partial u}{\partial Z} &= \frac{\partial u_\infty}{\partial Z} + \left(\frac{\partial V_{n_\infty}}{\partial Z} - \frac{\partial V_n}{\partial Z}\right) e_2 + (V_{n_\infty} - V_n)\frac{\partial e_2}{\partial Z} \\
\frac{\partial v}{\partial Z} &= \frac{\partial V_\infty}{\partial Z} - \left(\frac{\partial V_{n_\infty}}{\partial Z} - \frac{\partial V_n}{\partial Z}\right) e_3 - (V_{n_\infty} - V_n)\, e_3
\end{aligned}\right\} \qquad \text{(A-15a)}$$

Note that in the above

$$\frac{\partial u_\infty}{\partial Z} = v_\infty \frac{\partial \phi}{\partial Z} \text{ and } \frac{\partial v_\infty}{\partial Z} = -u_\infty \frac{\partial \phi}{\partial Z}\,. \qquad \text{(A-15b)}$$

Since $V_{n_\infty} = w_\infty e_1 - u_\infty e_2 + v_\infty e_3$, we have that

$$\frac{\partial V_{n_\infty}}{\partial Z} = w_\infty \frac{\partial e_1}{\partial Z} - u_\infty \frac{\partial e_2}{\partial Z} + v_\infty \frac{\partial e_3}{\partial Z} - (v_\infty e_2 + u_\infty e_3)\frac{\partial \phi}{\partial Z}\,. \qquad \text{(A-15c)}$$

From the shock relations (2.7), we have that $p = p_\infty + V_{n_\infty}\rho_\infty(V_{n_\infty} - V_n)$ which implies that on $X = 1$

$$\frac{\partial p}{\partial Z} = \rho_\infty\left[(V_{n_\infty} - V_n)\frac{\partial V_{n_\infty}}{\partial Z} + V_{n_\infty}\left(\frac{\partial V_{n_\infty}}{\partial Z} - \frac{\partial V_n}{\partial Z}\right)\right] \qquad \text{(A-16)}$$

In order to obtain an expression for $\frac{\partial V_n}{\partial Z}$, we differentiate the equation of state $\rho = \rho(h,p)$ to obtain

$$\frac{\partial \rho}{\partial Z} = \mathcal{K}_1 \frac{\partial h}{\partial Z} + \mathcal{K}_2 \frac{\partial p}{\partial Z} \qquad \text{(A-17)}$$

Differentiating the first and last equations in (2.7), we obtain for points on $X = 1$ that

$$\frac{\partial \rho}{\partial Z} = \rho_\infty\left[\frac{\partial V_{n_\infty}}{\partial Z} - (V_{n_\infty}/V_n)\frac{\partial V_n}{\partial Z}\right]/V_n$$

$$\frac{\partial h}{\partial Z} = V_{n_\infty}\frac{\partial V_{n_\infty}}{\partial Z} - V_n \frac{\partial V_n}{\partial Z}$$

Substituting these and (A-16) into (A-17) and using (A-4), we obtain

$$\frac{\partial V_{n_\infty}}{\partial Z} - \frac{\partial V_n}{\partial Z} = \left\{ \frac{V_n \,(V_{n_\infty} - V_n) \left[\frac{1}{V_n^2} + \frac{1}{a^2} + \mathcal{K}_1 \left(\frac{1}{\rho_\infty} - \frac{1}{\rho} \right) \right]}{\frac{\rho}{\rho_\infty} \left(1 - \frac{V_n^2}{a^2} \right)} \right\} \frac{\partial V_{n_\infty}}{\partial Z} \tag{A-18}$$

Finally, we have that

$$\frac{\partial e_1}{\partial Z} = \left[(1 - e_1^2) \frac{\partial c_z}{\partial Z} - e_1 e_3 \frac{\partial (c_\phi / c)}{\partial Z} \right] e_2 \tag{A-19a}$$

$$\frac{\partial e_2}{\partial Z} = - \left[e_1 \frac{\partial c_z}{\partial Z} + e_3 \frac{\partial (c_\phi / c)}{\partial Z} \right] e_2^2 \tag{A-19b}$$

$$\frac{\partial e_3}{\partial Z} = - \left[e_1 e_3 \frac{\partial c_z}{\partial Z} - (1 - e_3^2) \frac{\partial (c_\phi / c)}{\partial Z} \right] e_2 \tag{A-19c}$$

Substituting (A-15a) and (A-16) into (A-14) and using (A-15c), (A-18) and (A-19) to eliminate the terms $\frac{\partial V_n}{\partial Z}$, $\frac{\partial V_{n_\infty}}{\partial Z}$, $\frac{\partial e_i}{\partial Z}$ (i=1,2,3), we obtain from the compatability condition (A-14) a partial differential equation of the form

$$C_1 \frac{\partial c_z}{\partial Z} + C_2 \frac{\partial (c_\phi / c)}{\partial Z} = \mathcal{R}_s \tag{A-20}$$

The quantities C_1, C_2, and $\mathcal{R}_s$ appearing in (A-20) are given in section 3.3.

In addition to (A-20), we also have

$$\frac{\partial c}{\partial Z} = c_z + c_\phi \frac{\partial \phi}{\partial Z} \tag{A-21}$$

and

$$\frac{\partial c_\phi}{\partial Z} - Y_\phi \frac{\partial c_z}{\partial Y} + Y_z \frac{\partial c_\phi}{\partial Y} = 0 \tag{A-22}$$

The first equation (A-21) is the chain rule for the Z derivative of the shock shape function $c(\phi,z)$. The second equation (A-22) expresses through the chain rule that $\frac{\partial^2 c}{\partial z \partial \phi} = \frac{\partial^2 c}{\partial \phi \partial z}$. When C_1 is nonzero the equations (A-20) - (A-22) are formally a hyperbolic system with Z the time-like direction and, therefore, can be numerically solved using a predictor-corrector method to determine c, c_z, and c_ϕ. With the shock geometry determined, the Rankine-Hugoniot conditions determine all the flow variables on X = 1. The procedure is described in detail in section 3.3.

The above procedure requires that the coefficient C_1 does not change sign in the calculation. We now show that for a perfect gas $C_1 > 0$ when $w \geq a$.[+] The proof uses the definitions following (3.9c) (see sec. 3.3) and the shock relations (2.7). Consider first the case of a finite strength shock; i.e., $V_{n_\infty} > V_n$, $a > V_n$, $p > p_\infty$, $\rho > \rho_\infty$. Substituting (2.7) and $\mathcal{K}_1 = -\rho/h$ into the definition of A_0, we obtain

$$A_0 = \frac{(p - p_\infty)(V_n V_{n_\infty} + a^2 p_\infty/p)}{\rho_\infty V_{n_\infty}^2 (a^2 - V_n^2)} \quad . \qquad [\text{perf}] \qquad \text{(A-23)}$$

This implies that $A_0 > 0$ and it follows from the definition of A_1 that

$$A_1 > \rho(\nu_s w_\infty - c_z V_{n_\infty})A_0 \quad . \qquad \text{(A-24)}$$

From the shock relations and $w \geq a$, we have that

$$\nu_s w_\infty - c_z V_{n_\infty} = \nu_s w - c_z V_n \geq \nu_s a - c_z V_n > 0. \qquad \text{(A-25)}$$

Using (A-24) and (A-25) in the definition of C_1 we obtain

$$C_1 > \{\rho(\nu_s w_\infty - c_z V_{n_\infty})^2 A_0 - (p - p_\infty)(\nu_s^2 - c_z^2)\}/\nu_s^2 \qquad \text{(A-26)}$$

[+] It is expected that the significance of $C_1 > 0$ for the real gas case is the same as for a perfect gas but this fact has not been established at this time. In actual calculations, the value of C_1 is monitored at all points in the flow. If a change of sign is detected the calculation is

Note that since $u - c_\phi v/c = c_z w - V_n \nu_s$,

$$\beta_0^2 a^2 = (\nu_s w - c_z V_n)^2 - (\nu_s^2 - c_z^2)(a^2 - V_n^2)$$

which implies that

$$(\nu_s w_\infty - c_z V_{n_\infty})^2 = (\nu_s w - c_z V_n)^2 \geq (\nu_s^2 - c_z^2)(a^2 - V_n^2). \qquad \text{(A-27)}$$

Using (A-27), (A-23), (A-26), and $\rho = {}_\infty V_{n_\infty}/V_n$, we obtain

$$C_1 > \frac{(\nu_s^2 - c_z^2)(p - p_\infty)}{\nu_s^2 \rho_\infty V_{n_\infty} V_n} \left\{ V_n V_{n_\infty} + a^2 \frac{p_\infty}{p} - V_n V_{n_\infty} \right\} > 0$$

which is the desired result. Consider now the case of a weak shock; i.e., $V_n = V_{n_\infty} = a = a_\infty$, $p = p_\infty$, $\rho = \rho_\infty$. The quantity A_0 can be rewritten using (A-23) and the first two equations in (3.12) (with $\gamma = \Gamma = \Gamma_\infty$) as

$$A_0 = \frac{2(V_{n_\infty}^2 + a_\infty)}{(\gamma+1) V_{n_\infty}^2} \qquad \text{[perf]}$$

which implies that $A_0 = 4/\gamma+1$ for $V_{n_\infty} = a_\infty$. Since $A_0 > 0$ and $(\nu_s w_\infty - c_z V_{n_\infty}) > 0$, it follows directly from the definitions that $A_1 > 0$ and hence $C_1 > 0$.

APPENDIX B

SYMMETRY CONDITIONS

In this appendix, the symmetry conditions (3.21) and (3.22) used in the symmetric problem are derived. Recall that in this problem all flow variables are even functions of ϕ about $\phi = 0$ and π in the physical space except v which is an odd function of ϕ about $\phi = 0$ and π. Also, the body geometry and shock geometry functions are even functions of ϕ about 0 and π. In this appendix, ϕ^* will denote either 0 or π. To avoid any confusion we specify that: a function $u(r,\phi,z)$ is <u>even</u> about ϕ^* if and only if $u(r,\phi^*+\tau,z) = u(r,\phi^*-\tau,z)$ for all τ such that $\phi^* + \tau$ is in the domain of definition of u; a function $u(r,\phi,z)$ is odd about ϕ^* if and only if $u(r,\phi^*+\tau,z) = - u(r,\phi^*-\tau,z)$.

Consider first the effect of the mapping (3.1) on a quantity u which in the physical plane is given by an even (or odd) function of ϕ about $\phi = \phi^*$. Let u be given by $u(r,\phi,z)$ in the physical space and $\bar{u}(\bar{x},\bar{y},\bar{z})$ in the $(\bar{x},\bar{y},\bar{z})$ space; further, let $\bar{y}^* = \phi^*/\phi_o$ (i.e., $\bar{y}^* = 0$ or 1). Note that from (3.1), $\bar{x}$ is an even function of ϕ about $\phi = \phi^*$ for fixed r and z. Also, inverting (3.1), we have

$$r(\bar{x},\bar{y},\bar{z}) = \bar{x}[c(\bar{z},\phi_o\bar{y}) - b(\bar{z},\phi_o\bar{y})] + b(\bar{z},\phi_o\bar{y})$$

which is an even function of $\bar{y}$ about $\bar{y}^*$ for fixed $\bar{x},\bar{y}$. It therefore follows from

$$u(r,\phi,z) = \bar{u}(\bar{x}(r,\phi,z),\phi/\phi_o,z)$$

and

$$\bar{u}(\bar{x},\bar{y},\bar{z}) = u(r(\bar{x},\bar{y},\bar{z}),\bar{y}\phi_o,\bar{z})$$

that $u(r,\phi,z)$ is an even (odd) function of ϕ about $\phi = \phi^*$ for fixed r,z if and only if $\bar{u}(\bar{x},\bar{y},\bar{z})$ is an even (odd) function of $\bar{y}$ about $\bar{y}^*$ for fixed $\bar{x},\bar{z}$.

Consider now the quantity u given in the computational space (X,Y,Z) by the function U(X,Y,Z). Note that $U(X,Y,Z) = \bar{u}(f(X,Y,Z), g(Y,Z),Z)$. For any fixed Z, let Y^* denote the value of Y such that $\bar{y}^* = g(Y^*,Z)$ (i.e., Y^* is 0 or 1). The Taylor development of U(X,Y,Z) about (X,Y^*,Z) where X and Z are fixed gives

$$U(X,Y^* \pm \Delta Y,Z) = U(X,Y^*,Z) \pm \Delta Y\, U_Y(X,Y^*,Z)$$

$$+ \frac{(\Delta Y)^2}{2} U_{YY}(X,Y^*,Z) \pm \frac{(\Delta Y)^3}{3!} U_{YYY}(X,Y^*,Z) + O(\Delta Y)^4.$$

Adding and subtracting the above expressions, we obtain

$$U(X,Y^* \pm \Delta Y,Z) - U(X,Y^* \mp \Delta Y,Z) = \pm 2\, \Delta Y\, U_Y(X,Y^*,Z) + O(\Delta Y)^3 \qquad (B.1)$$

and

$$U(X,Y^* \pm \Delta Y,Z) + U(X,Y^* \mp \Delta Y,Z) = 2\, U(X,Y^*,Z) + \Delta Y^2 U_{YY}(X,Y^*,Z)$$

$$+ O(\Delta Y)^4. \qquad (B.2)$$

For use in the above we have, by the chain rule, that

$$U_Y = \bar{u}_{\bar{x}} f_Y + \bar{u}_{\bar{y}} g_Y \qquad (B.3)$$

and

$$U_{YY} = \bar{u}_{\bar{x}\bar{x}} (f_y)^2 + 2\bar{u}_{\bar{x}\bar{y}} f_Y g_Y + \bar{u}_{\bar{y}\bar{y}} (g_y)^2$$

$$+ \bar{u}_{\bar{x}} f_{YY} + \bar{u}_{\bar{y}} g_{YY} \qquad (B.4)$$

Suppose that $u(r,\phi,z)$ is an even function of ϕ about ϕ^*. By our previous remarks, it follows that $\bar{u}(\bar{x},\bar{y},\bar{z})$ is an even function of $\bar{y}$ about $\bar{y}^*$. Since $\bar{u}$ is even about $\bar{y}^*$, we have that $\bar{u}_{\bar{y}}(\bar{x},\bar{y}^*,\bar{z}) = 0$. Also, if we assume that $f_Y(X,Y^*,Z) = 0$, it follows from (B.3) that $U_Y(X,Y^*,Z) = 0$. This implies, using (B.1), that

$$U(X,Y^* \pm \Delta Y,Z) = U(X,Y^* \mp \Delta Y,Z) + O(\Delta Y)^3$$

which is the same as (3.21).

Suppose now that $u(r,\phi,z)$ is an odd function of ϕ about ϕ^*. By our previous remarks, $\bar{u}(\bar{x},\bar{y},\bar{z})$ is an odd function of $\bar{y}$ about $\bar{y}^*$. Hence, it follows that

$$\bar{u}(\bar{x},\bar{y}^*,\bar{z}) = \bar{u}_{\bar{y}}(\bar{x},\bar{y}^*,\bar{z}) = 0.$$

Since the above holds for all $\bar{x}$, we also have that $\bar{u}_{\bar{x}}(\bar{x},\bar{y}^*,z) = \bar{u}_{\bar{x}\,\bar{x}}(\bar{x},\bar{y}^*,\bar{z}) = 0$. If $f_Y(X,Y^*,Z) = 0$, it follows from (B.4) that

$$U_{YY}(X,Y^*,Z) = \bar{u}_{\bar{y}}(\bar{x},\bar{y}^*,\bar{z})g_{YY}(Y^*,Z)$$

and from (B.3) that

$$\bar{u}_{\bar{y}}(\bar{x},\bar{y}^*,\bar{z}) = U_Y(X,Y^*,Z)/g_Y(Y^*,Z)$$

since $U(X,Y^*,Z) = 0$, it follows using (B.2) that

$$U(X,Y^*\pm\Delta Y,Z) + U(X,Y^*\mp\Delta Y,Z) =$$
$$\Delta Y^2 U_Y(X,Y^*,Z)g_{YY}(Y^*,Z)/g_Y(Y^*,Z) + 0(\Delta Y)^4$$

Using (B.1) to evaluate U_Y in the above, we obtain

$$U(X,Y^*\pm\Delta Y,Z)[2\mp\Delta Y g_{YY}(Y^*,Z)/g_Y(Y^*,Z)] =$$
$$-U(X,Y^*\pm\Delta Y,Z)[2\pm\Delta Y g_{YY}(Y^*,Z)/g_Y(Y^*,Z)] + 0(\Delta Y)^4$$

which is the same as (3.22).

APPENDIX C

CFL CONDITIONS

In this appendix, we derive a necessary stability condition used for determining the step size ΔZ associated with a given computational mesh $\Delta X, \Delta Y$. The derivation is in the geometric spirit of CFL and consists of comparing the domain of dependence* of the difference equations, $\mathcal{D}_{d.e.}$, to the domain of dependence of the partial differential equations, $\mathcal{D}_{p.d.e.}$. Following CFL, it is necessary to restrict the largest value of ΔZ so that

$$\mathcal{D}_{p.d.e} \subset \mathcal{D}_{d.e.} \tag{C.1}$$

for all computational points. For the quasi-linear system of hyperbolic equations considered here this condition is generally regarded as a necessary (but not always sufficient) condition for numerical stability. Indeed, it has been observed that the MacCormack scheme exhibits numerical instability for particular first order systems even when a geometric CFL stability condition is obeyed (ref. 12). For our system no such instabilities have ever been demonstrated nor observed in calculations.

*In this appendix, the domain of dependence of a point, 0, with Z coordinate $Z_o + \Delta Z$, is understood to be the smallest closed region of the plane $Z \stackrel{o}{=} Z_o$, $\mathcal{D}$, such that the initial data outside of $\mathcal{D}$ do not influence the solution at 0.

Consider an arbitrary interior computational point, 0, with coordinates $(Z_o + \Delta Z, X_o, Y_o)$. The domain of dependence of 0 for the finite difference schemes given by (3.6a) and (3.6b) depends on the parameter j(= 0 or 1). These are depicted in Fig. C-1. The dots in the figures represent the only computational points which can influence the numerical solution at 0 for one complete (predictor and corrector) step. In stability considerations, it is necessary to consider the limit for successive mesh requirements with $\Delta X/\Delta Z$ and $\Delta Y/\Delta Z$ held fixed. In this limit, the points in and on the boundary of the shaded regions are the only ones that can influence the numerical solution at 0.

Consider now the domain of dependence of 0 for the system of partial differential equations in the computational space, see (3.3). For the purposes of stability analysis it is sufficient to consider a quasi-linear system equivalent to (3.3) given by

$$\mathcal{A}\frac{\partial Q}{\partial Z} + \mathcal{B}\frac{\partial Q}{\partial X} + \mathcal{C}\frac{\partial Q}{\partial Y} = \mathcal{R} \qquad (C.2)$$

where

$$Q = \begin{pmatrix} p \\ u \\ v \\ w \end{pmatrix}$$

In the above,

$$\mathcal{A} = \mathcal{O}\left[\frac{\partial U}{\partial Q}\right], \quad \mathcal{B} = \mathcal{O}\left\{X_z\left[\frac{\partial U}{\partial Q}\right] + X_r\left[\frac{\partial \mathcal{F}}{\partial Q}\right] + X_\phi\left[\frac{\partial \mathcal{G}}{\partial Q}\right]\right\},$$

$$\mathcal{C} = \mathcal{O}\left\{Y_z\left[\frac{\partial U}{\partial Q}\right] + Y_\phi\left[\frac{\partial \mathcal{G}}{\partial Q}\right]\right\}$$

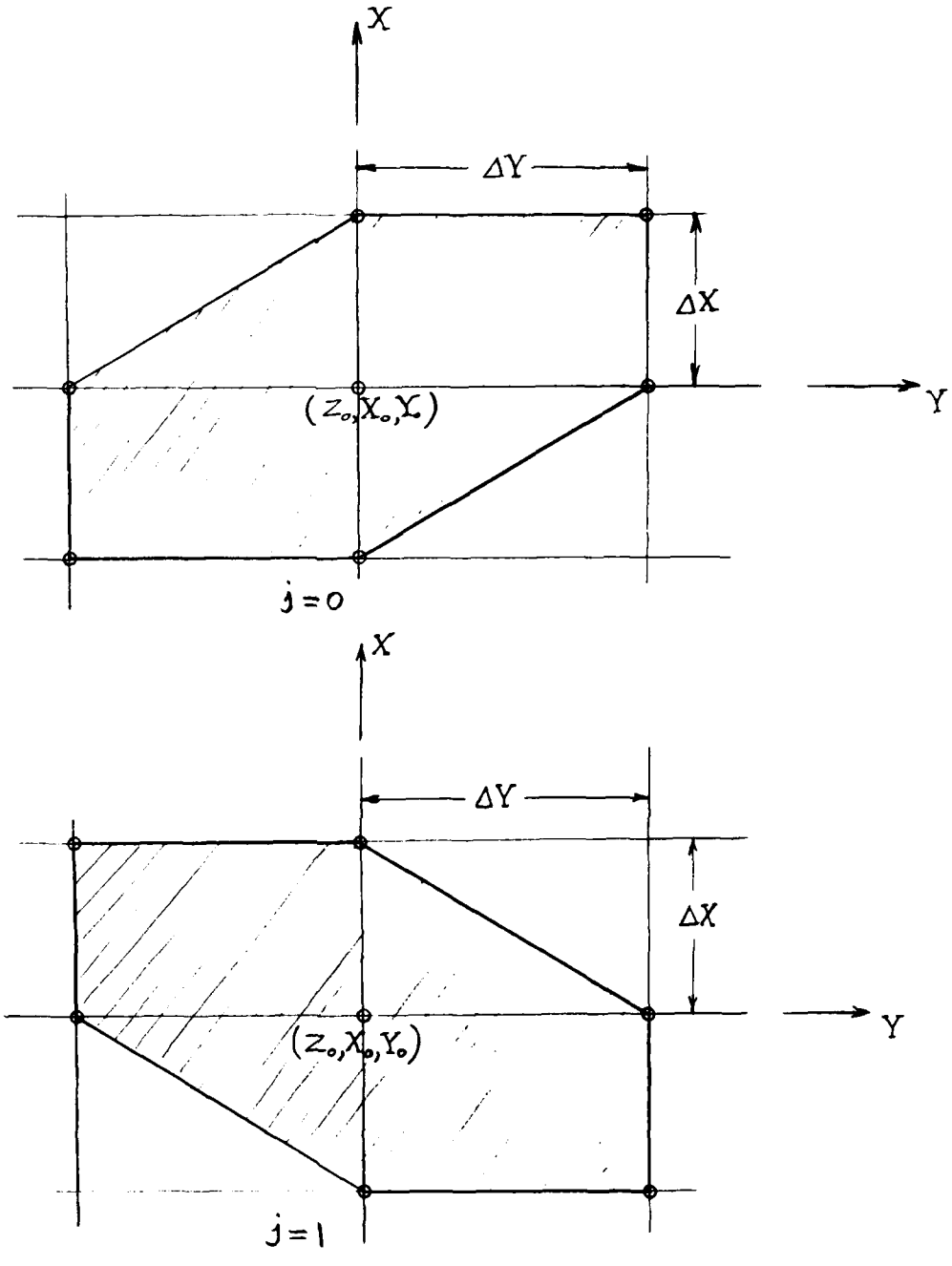


Fig. C-1. Domain of dependence for the interior finite difference equations

where $[\frac{\partial U}{\partial Q}]$, $[\frac{\partial \mathcal{F}}{\partial Q}]$, and $[\frac{\partial \mathcal{G}}{\partial Q}]$ are Jacobian matrices and $\mathcal{O}$ is the non-singular matrix given in Appendix A. The matrices $\mathcal{A}$ and $\mathcal{B}$ are given by (A-5a) and (A-5b) respectively; the matrix $\mathcal{C}$ is given by

$$\mathcal{C} = \begin{bmatrix} \mathcal{K}_2 B & -\mathcal{K}_1 uB & \frac{\rho}{r} Y_\phi - vB\mathcal{K}_1 & Y_z \rho - w\mathcal{K}_1 B \\ Y_z & 0 & 0 & \rho B \\ 0 & \rho B & 0 & 0 \\ \frac{1}{r} Y_\phi & 0 & \rho B & 0 \end{bmatrix}$$

The inhomogeneous term $\mathcal{R}$ is not material to the present analysis and will not be given. The domain of dependence of the system (3.3) is the same as that of (C.2). The domain of dependence of 0 for the system (C.2) is contained in the closed region formed by the intersection of the characteristic conoid with vertex at 0 and the plane $Z = Z_o$ (c.f.; ref. 10, pp. 649-652).

The pertinent facts concerning the geometry of characteristic conoids associated with systems of the type (C.2) will be briefly reviewed here; for a more detailed explanation see ref. 10 pp. 577-599. The characteristic conoid with vertex 0 is the envelope of all characteristic surfaces through 0. A surface $\psi(Z,X,Y) = 0$ is characteristic at a point if its normal at the point satisfies the characteristic condition

$$\mathcal{H}(\lambda_1,\lambda_2,\lambda_3;\ Q;\ X,Y,Z) \equiv \det(\lambda_1 \mathcal{A} + \lambda_2 \mathcal{B} + \lambda_3 \mathcal{C}) = 0 \qquad (C.3)$$

where $\lambda_1 = \frac{\partial\psi}{\partial Z}$, $\lambda_2 = \frac{\partial\psi}{\partial X}$, and $\lambda_3 = \frac{\partial\psi}{\partial Y}$. The surface of the characteristic conoid is generated by curves, called rays or bicharacteristics which are the lines of contact between the characteristic surfaces and the conoid they envelope. These curves, or rays, are given by the ordinary differential equations

$$\frac{dZ}{ds} = \frac{\partial \mathcal{H}}{\partial \lambda_1}, \quad \frac{dX}{ds} = \frac{\partial \mathcal{H}}{\partial \lambda_2}, \quad \frac{dY}{ds} = \frac{\partial \mathcal{H}}{\partial \lambda_3} \tag{C.4}$$

where s is a parameter. Each ray through 0 is determined by selecting real values for λ_2 and λ_3 and determining λ_1 by satisfying the characteristic condition (C.3) at 0. In general, when $\mathbb{A}$, $\mathbb{B}$, and $\mathbb{C}$ are not constant, λ_2, λ_3, and λ_1 vary along the ray. When (C.2) is a quasi-linear system like (C.2) the rays, conoids, and domains of dependence will depend on the solution vector Q. In the case when the coefficient matrices $\mathbb{A}$, $\mathbb{B}$, and $\mathbb{C}$ are constant λ_1, λ_2, and λ_3 satisfying (C.3) are constant along each ray. The rays generating the conoid are straight lines since the right hand sides of (C.4) are constant. The characteristic surfaces in this case are hyperplanes and cones formed by their envelopes.

In the present analysis $\Delta Z, \Delta X, \Delta Y$ are assumed to be small, thus the domain of dependence of $0 = (Z_o + \Delta Z, X_o, Y_o)$ can be approximated by considering the matrices $\mathbb{A}$, $\mathbb{B}$, and $\mathbb{C}$ as constants with elements evaluated at the point (Z_o, X_o, Y_o). For the remainder of this appendix, it will be understood without changing notation that all quantities appearing in these matrices are held fixed at their values at (Z_o, X_o, Y_o).

Making the substitutions indicated above, the characteristic condition (C.3) for the system of equations (C.2) is

$$\left.\begin{aligned} \mathcal{H} &= \mathcal{H}_1(\lambda_1,\lambda_2,\lambda_3)\mathcal{H}_2(\lambda_1,\lambda_2,\lambda_3) = 0 \\ &\text{where} \\ \mathcal{H}_1(\lambda_1,\lambda_2,\lambda_3) &= \lambda_1 w + \lambda_2 A + \lambda_3 B \\ \mathcal{H}_2(\lambda_1,\lambda_2,\lambda_3) &= \mathcal{H}_1^2 - [(\lambda_1 + \lambda_2 X_z + \lambda_3 Y_z)^2 + \lambda_2^2 X_r^2 + (\lambda_2 X_\phi + \lambda_3 Y_\phi)^2 \frac{1}{r^2}]a^2 \end{aligned}\right\} \quad (C.5)$$

The ray cone therefore has two sheets, one corresponding to $\mathcal{H}_1 = 0$ and one corresponding to $\mathcal{H}_2 = 0$. The rays generating the sheet corresponding to $\mathcal{H}_1 = 0$ are given, using (C.4), by

$$\frac{dZ}{ds} = w, \quad \frac{dX}{ds} = A, \quad \frac{dY}{ds} = B.$$

Hence the sheet corresponding to $\mathcal{H}_1 = 0$ is a degenerate cone consisting of a single ray through 0. The domain of dependence for this sheet is a point in the plane $Z = Z_o$ with coordinates $X = X_o - (A/W)\Delta Z$ and $Y = Y_o - (B/W)\Delta Z$. This ray corresponds in the physical space (z,ϕ,r) to a streamline. The sheet corresponding to $\mathcal{H}_2 = 0$ is a true cone which corresponds in physical space to the Mach cone.** In physical

**A characteristic surface $\psi(Z,X,Y) = 0$ can be represented by $\bar{\psi}(z,r,\phi) = 0$ in physical space. By the chain rule

$$\bar{\psi}_z = \lambda_1 + \lambda_2 X_z + \lambda_3 Y_z, \quad \bar{\psi}_r = \lambda_2 X_r, \text{ and } \bar{\psi}_\phi = \lambda_2 X_\phi + \lambda_3 Y_\phi$$

since $\lambda_1 = \psi_Z$, $\lambda_2 = \psi_X$ and $\lambda_3 = \psi_Y$. It follows that

$$\mathcal{H}_2(\lambda_1,\lambda_2,\lambda_3) = (\nabla\bar{\psi}\cdot\vec{q})^2 - |\nabla\bar{\psi}|^2 a^2 = 0$$

where $\vec{q}$ is the velocity vector. This equation indicates that in (z,r,ϕ) space the cosine of the angle between the streamline and the normals to the characteristic surfaces enveloping the cone associated with $\mathcal{H}_2 = 0$ is $\pm a/|\vec{q}| = \pm M^{-1}$.

space the streamline through the Mach cone vertex is interior to the cone. Since a continuous transformation cannot change this situation, it follows that in the computational space the cone corresponding to $\mathcal{H}_2 = 0$ contains the ray corresponding to $\mathcal{H}_1 = 0$. Therefore the domain of dependence of (C.2) is determined entirely by the sheet $\mathcal{H}_2 = 0$.

The CFL stability condition for each of the MacCormack schemes is illustrated in Fig. C-2. The shaded area is the region formed by the intersection on the plane $Z = Z_o$ of the cone with vertex at 0 corresponding to $\mathcal{H}_2 = 0$. Condition (C.1) for j = 0 or 1 is satisfied if and only if

$$\max_{i=1,2} (\ell_i) \leq \Delta X, \quad \max_{i=3,4} (\ell_i) \leq \Delta Y, \quad \text{and} \quad \max_{i=5,6} (\ell_i^j) \leq \Delta X \Delta Y \Big/ \sqrt{\Delta X^2 + \Delta Y^2} \tag{C.6}$$

In the above, ℓ_i (i=1,$\cdots$,6) are projections of the base of the cone in various directions as indicated in Fig. C-2. These distances depend on the value of ΔZ (recall that ΔX and ΔY are assumed fixed in the present analysis).

Consider now the determination of the projection of the cone's base on any directed line in the $Z = Z_o$ plane with, say, X and Y direction numbers σ_x and σ_y, respectively. Since any characteristic plane $\psi(Z,X,Y) = 0$ through 0 is tangent to the cone, the particular ones which have normals with $\psi_X = \lambda_2 = \sigma_x$ and $\psi_Y = \lambda_3 = \sigma_y$ intersect the plane $Z = Z_o$ on lines which are both tangent to the cone's base and normal to the direction (σ_x,σ_y). The situation is shown in Fig. C-3. The point of tangency, Q, is the intersection on the $Z = Z_o$ plane of the bicharacteristic ray with $\lambda_2 = \sigma_x$, $\lambda_3 = \sigma_y$, and $\lambda_1 = \sigma_z$ where σ_z is the solution of

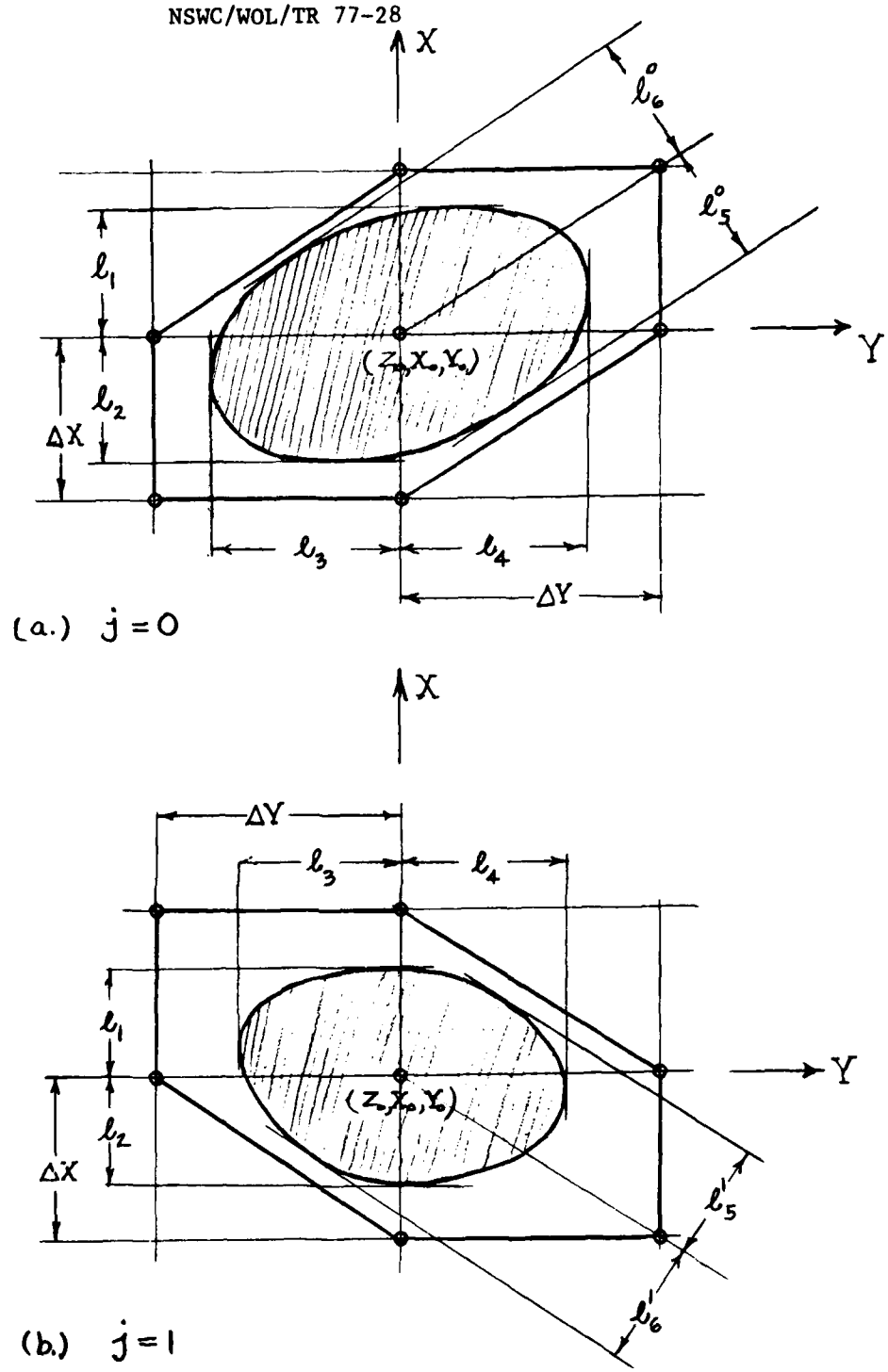


Fig. C-2. CFL conditions for MacCormack schemes

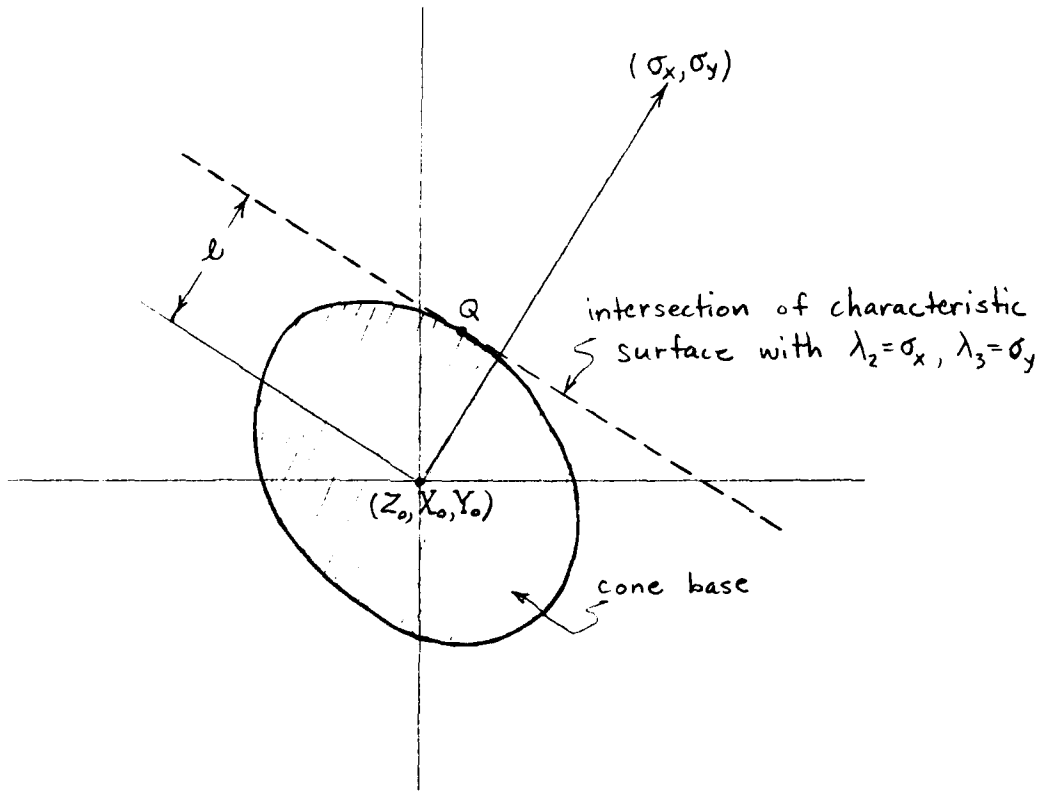

(σ_x, σ_y)
ℓ
Q
intersection of characteristic surface with $\lambda_2=\sigma_x$, $\lambda_3=\sigma_y$
(Z_o, X_o, Y_o)
cone base

Fig. C-3.

$\mathcal{H}_2(\sigma_z,\sigma_x,\sigma_y) = 0$. For $w > a$, there are precisely two distinct solutions of this equation. The two values of σ_z determine two rays and, hence, two points of tangency for each direction (σ_x,σ_y). The X,Y coordinates of Q for each value of σ_z are determined by integrating (C-4) from $Z = Z_o + \Delta Z$ to $Z = Z_o$; i.e.,

$$X - X_o = -\left(\frac{\partial \mathcal{H}_2}{\partial \lambda_2} \Big/ \frac{\partial \mathcal{H}_2}{\partial \lambda_1}\right) \Delta Z, \quad Y - Y_o = -\left(\frac{\partial \mathcal{H}_2}{\partial \lambda_3} \Big/ \frac{\partial \mathcal{H}_2}{\partial \lambda_1}\right) \Delta Z$$

where $\frac{\partial \mathcal{H}_2}{\partial \lambda_i}$ $(i=1,2,3)$ are evaluated at $\lambda_1 = \sigma_z$, $\lambda_2 = \sigma_x$, $\lambda_3 = \sigma_y$. The projected distance ℓ in Fig. (C.3), is given by

$$\ell = \left|\sigma_x (X - X_o) + \sigma_y (Y - Y_o)\right| \Big/ \sqrt{\sigma_x^2 + \sigma_y^2} \ .$$

Since $\mathcal{H}_2(\lambda_1,\lambda_2,\lambda_3)$ is homogeneous in $\lambda_1,\lambda_2,\lambda_3$ it follows that $\lambda_1 \frac{\partial \mathcal{H}_2}{\partial x_1} + \lambda_2 \frac{\partial \mathcal{H}_2}{\partial \lambda_2} + \lambda_3 \frac{\partial \mathcal{H}_2}{\partial \lambda_3} = 0$ when $\mathcal{H}_2(\lambda_1,\lambda_2,\lambda_3) = 0$. It therefore follows that

$$\ell = |\sigma_z| \Delta Z \Big/ \sqrt{\sigma_x^2 + \sigma_y^2} \ . \tag{C.7}$$

The distances ℓ_1 and ℓ_2 are projections on the direction $\sigma_x = 1$, $\sigma_y = 0$ (c.f., Fig. C-2). Solving $\mathcal{H}_2(\sigma_z,1,0) = 0$, we obtain using (C-7) that

$$\ell_{1,2} = \left| X_z a^2 - AW \pm a \sqrt{(A - W X_z)^2 + (W^2 - a^2)\left(X_r^2 + \frac{1}{r^2} X_\phi^2\right)} \right| \frac{\Delta Z}{(w^2 - a^2)} \ .$$

Substituting the above into the first inequality in (C.6), we obtain

$$\Delta Z \leq \frac{(w^2 - a^2)}{\mu_1} \Delta X \tag{C.8}$$

where μ_1 is given in section 3.6. The distances ℓ_3 and ℓ_4 are projections on the direction $\sigma_x = 0$, $\sigma_y = 1$. Hence ℓ_3 and ℓ_4 are given by (C-7) with σ_z a solution of $H_2(\sigma_z,0,1) = 0$; i.e.,

$$\ell_{3,4} = \left| Y_z\, a^2 - wB \pm a\sqrt{(Y_\phi^2/r^2)(w^2 + v^2 - a^2)} \right| \frac{\Delta Z}{(w^2 - a^2)}$$

Substituting this into the second inequality in (C.6), we obtain

$$Z \leq \frac{(w^2 - a^2)}{\mu_2} \Delta X \tag{C.9}$$

where μ_2 is given in section 3.6. The distances ℓ_5^j and ℓ_6^j are projections on the direction $\sigma_x = \Delta Y$, $\sigma_y = -(-1)^j \Delta X$. Hence ℓ_5^j and ℓ_6^j are given by (C-7) with σ_z a solution of $H_2(\sigma_z, \Delta Y, -(-1)^j \Delta X) = 0$; i.e.,

$$\ell_{5,6}^j = \frac{\Delta Y \Delta Z}{\left(\sqrt{\Delta X^2 + \Delta Y^2}\right)(w^2 - a^2)} \left\{ \delta(wB - Y_z\, a^2) - (wA - X_z\, a^2) \right.$$

$$\left. \pm a\sqrt{(w^2 - a^2)\left[X_r^2 + \frac{1}{r^2}(X_\phi - \delta Y_\phi)^2\right] + (wX_z - A + \delta v Y_\phi / r)^2} \right\}$$

where

$$\delta = (-1)^j (\Delta X / \Delta Y)$$

Substituting this into the last inequality in (C.6), we obtain

$$\Delta Z \leq \frac{(w^2 - a^2)}{\mu_3} \Delta X \tag{C.10}$$

where μ_3 is given in section 3.6.

For any interior computational point $(Z_o + \Delta Z, X_o, Y_o)$, the largest value of ΔZ for which the CFL condition (C.1) is satisfied is

$$\Delta Z = \left\{ (w^2 - a^2)/\max(\mu_1, \mu_2, \mu_3) \right\} \Delta X \qquad (C.11)$$

where the terms in the bracket are evaluated at (Z_o, X_o, Y_o). It remains now to consider the points on the boundary of the computational domain. The computational scheme for points on the boundaries $Y = 0$ and $Y = 1$ which are not on $X = 0$ or $X = 1$ is essentially the same as the interior point scheme (c.f., section 3.5); hence, the CFL condition for such points is the same as for interior points. The domain of dependence of the partial differential equations for points on the boundaries $X = 0$ and $X = 1$ is essentially as described above for interior points except that only the portion of the characteristic cone's base lying in $0 \leq X \leq 1$ is considered. The domain of dependence of the difference schemes used on the boundaries $X = 0$ and $X = 1$ are illustrated in Fig. C-4 and C-5. Note that at the boundary $X = 0$ the domain of dependence of the scheme when $j = 0$ is used for interior points depends on whether the second order option (c.f., eq. (3.19)) is used. Comparing Figs. C-4 and C-5 with C-2, we see that the CFL condition (C.1) for the points on the boundaries $X = 0$ or 1 is satisfied if ΔZ is chosen as if these points were interior points (i.e., using (C.11)). Therefore in order to insure that (C.1) is satisfied for all the computational points, we take the smallest ΔZ obtained from (C.11).

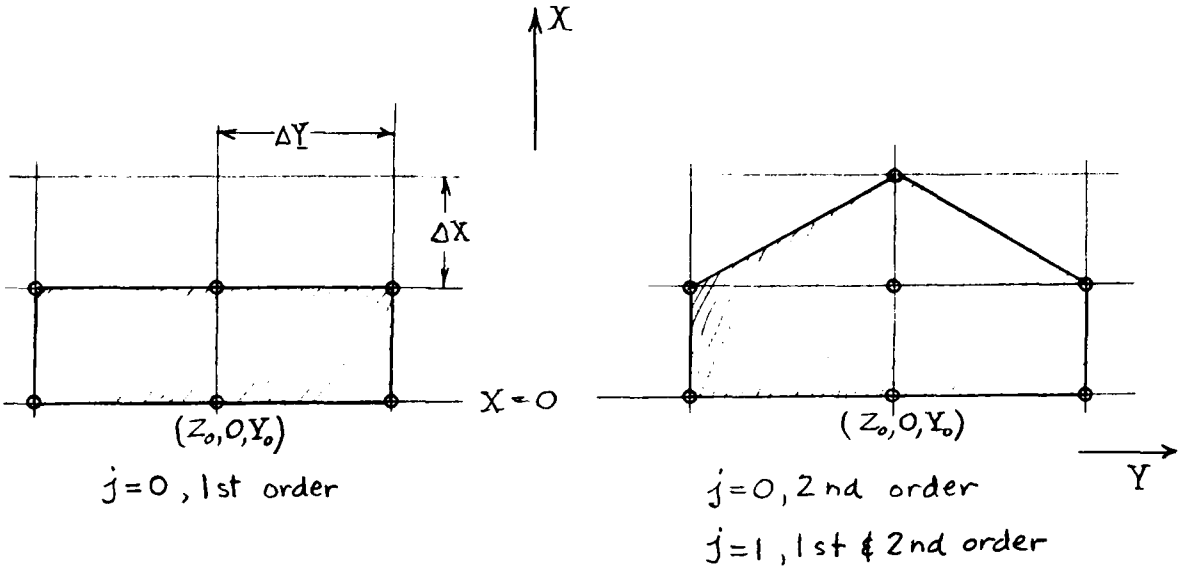


Fig. C-4. Domains of dependence of numerical scheme for wall points (X = 0)

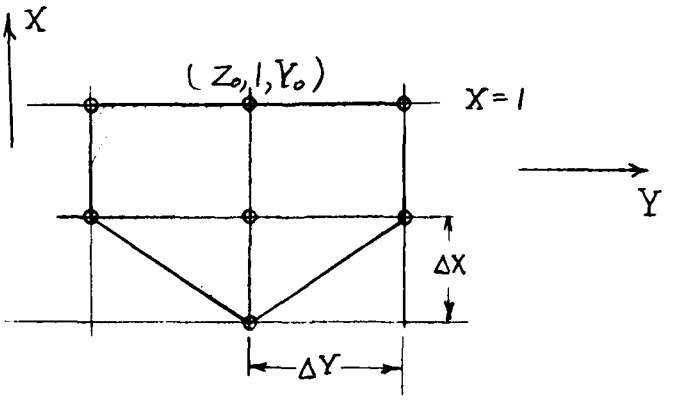


Fig. C-5. Domain of dependence of numerical scheme for bow shock points (X = 1)

6. REFERENCES

1. Kutler, P., Reinhardt, W. A., and Warming, R. F., "Multishocked, Three-Dimensional Supersonic Flowfields with Real Gas Effects," AIAA Jour., Vol. 11, No. 5, 1973, pp. 657-664.

2. MacCormack, R. W., "The Effect of Viscosity in Hypervelocity Impact Cratering," AIAA Paper 69-354, Cincinnati, Ohio, April 30 - May 2, 1969.

3. Thomas, P. D., Vinokur, M., Bastianon, R., and Conti, R. J., "Numerical Solution for Three-Dimensional Inviscid Supersonic Flow," AIAA Jour., Vol. 10, No. 7, 1972, pp. 887-894.

4. Landau, L. D. and Lifshitz, E. M., Fluid Mechanics, Pergumon Press (Addison-Wesley), 1959.

5. Ames Research Staff, "Equations, Tables, and Charts for Compressible Flow," NACA Rep. 1135, 1953.

6. Kyriss, C. L. and Harris, T. B., "A Three-Dimensional Flow Field Computer Program for Maneuvering and Ballistic Re-entry Vehicles," Tenth USN Sym. on Aeroballistics, July 15 - 17, 1975.

7. Chaussee, D. S., Holtz, T. and Kutler P., "Inviscid Supersonic/ Hypersonic Body Flowfields and Aerodynamics from Shock-Capturing Technique Calculations," AIAA Paper No. 75-837, Hartford, Conn. June 16 - 18, 1975.

8. Moretti, G. and Pandolfi, "Analysis of the Inviscid Flow About a Yawed Cone, Preliminary Studies," PIBAL Rep. No. 72-18, 1972

9. Marconi, F. and Salas, M., "Computation of Three Dimensional Flows About Aircraft Configurations," Computers and Fluids, Vol. 1, pp. 185-195, 1973.

10. Courant, R. and Hilbert, D., Methods of Mathematical Physics, Vol. II, Interscience Publishers, New York, 1962.

11. Kentzer, C. P., "Discretization of Boundary Conditions on Moving Discontinuities, "Second International Conf. on Num. Methods in Fluid Dynamics, Univ. of Calif., Berkeley, Sept. 15 - 19, 1970.

12. Turkel, E., "Phase Error and Stability of Second Order Methods for Hyperbolic Problems I," Journal of Computational Physics Vol. 15 pp. 226-250, 1974.

PART II: PROGRAMMING

7. GENERAL REMARKS

This part of the report contains a description of a FORTRAN program based on the numerical methods discussed in Part I.

The program consists of a main or executive routine, referred to as MAIN*, and several subroutines. MAIN contains the complete program structure and controls all aspects of the calculation. Most of the actual operations, however, are carried out in the subroutines many of which are called directly from MAIN. For the purpose of discussion, the subroutines are grouped into three functional types; subroutines used in the flow field calculation (see Sec. 10), auxiliary subroutines (see Sec. 11), and input-output subroutines (see Sec. 12). As a general rule, the transfer of arguments between the various subroutines and MAIN is via a COMMON block of variables. A table (see Sec. 8) is included which gives for each variable in COMMON, its description in terms of Part I notation (where possible), the primary routine in which it is defined, and the name of its COMMON block. The two COMMON blocks CSERCH and SAVRG, being used only in the real gas subroutines RGAS, HRGAS, and SERCH, are not included in this table. MAIN and all subroutines are described in Sections 9-12 which follow. These descriptions are intended to serve as a guide to the use of the FORTRAN listings which can be found in Appendix D.

In the following sections, we will occasionally refer to the second volume of this work.** This will be referred to herein as the "User's Manual".

*To distinguish between subroutine names and FORTRAN arrays or variable names, subroutine names are capitalized and underlined.

**Solomon, J. M., Ciment, M., Ferguson, R. E., Bell, J. B., and Wardlaw, A. B., A Program for Computing Steady Inviscid Three-Dimensional Supersonic Flow on Reentry Vehicles; Vol. II: User's Manual, NSWC/WOL/TR 77-32

8. COMMON

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
ACH	M_∞	INPUT	COUT
ALNS	α_n = ACONE(1), see fig. 8 of User's Manual	BODYR	CBENT
ASQ(N,M)	$a^2(X_n, Y_m)$	DECODE	BLANK
ATTA	α	INPUT	COUT
AUQN	parameter used for bent cone calcs.	BODY BODYR	CBENT
AX	a, sound speed	RGAS	RGASS
A2(J)	$\sqrt{1 + b_z^2 + (b_\phi/b)^2} = \nu_w$, (1)[+]	BODYP	BLKØ3[++]
A3(J)	$[b_{\phi\phi}/b - (b_\phi/b)^2]Y_\phi$, (1)	BODYP	BLKØ3
A4(J)	$b_{z\phi}/b - (Y_z/Y_\phi)[(b_{\phi\phi}/b) - (b_\phi/b)^2] - (b_z/b)(b_\phi/b)$, (1)	BODYP	BLKØ3
A5(J)	$b_{z\phi}/Y_\phi$, (1)	BODYP	BLKØ3
A7(J)	$[b_{zz} - b_{z\phi}(Y_z/Y_\phi)]$, (1)	BODYP	BLKØ3
B(M)	$b(Y_m)$	BODY	CBODY
BETA	parameter used for bent cone calcs.	BODY	CBENT
BJ	$-\delta$	MAIN	CEVAL
BPHI(M)	$b_\phi(Y_m)$	BODY	CBODY
BPHIO(M)	$b_\phi(Y_m)$ at previous step	MAIN	CBODYP
BPHIT(J)	temporary storage for b_ϕ, (1)	BODYP	CBODYP
BPHPHI	$b_{\phi\phi}$	BODY	CBODY
BPHPHO(M)	$b_{\phi\phi}(Y_m)$ at previous step	MAIN	CBODYP

[+]Numbers in parenthesis refer to the remarks appearing at the end on this list.

[++]The symbol Ø here is used to denote zero.

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
BPHPHT(J)	temporary storage for $b_{\phi\phi}$, (1)	MAIN	CBODYP
BSN	sin(ALNS)	BODYR	CBENT
BZ(M)	$b_z(M)$	BODY	CBODY
BZO(M)	$b_z(M)$ at previous step	MAIN	CBODYP
BZPHI	$b_{z\phi}$	BODY	CBODY
BZPHIO(M)	$b_{z\phi}(Y_m)$ at previous step	MAIN	CBODYP
BZPHIT(J)	temporary storage for $b_{z\phi}$, (1)	BODYP	CBODYP
BZT(J)	temporary storage for b_z, (1)	BODYP	CBODYP
BZZ	b_{zz}	BODY	CBODY
BZZO(M)	$b_{zz}(Y_m)$ at previous step	MAIN	CBODYP
BZZT(J)	temporary storage for b_{zz}, (1)	BODYP	CBODYP
C(M)	$c(Y_m)$	MAIN	BLANK
CE(I,N,L)	E, (2) and (3)	EVAL	BLKØ1
CENUF	parameter used for bent cone calcs.	INPUT	CBENT
CF(I,N,L)	F, (2) and (3)	EVAL	BLKØ1
CFL	stability factor; i.e., $\max\{\mu/(w^2 - a^2)\}$	EVAL	CEVAL
CG(I,N,J)	G, (1) and (2)	EVAL, EVALSY	BLKØ1
COSBN	COS(THETABN)	BODYR	CBENT
COSPHI(M)	$\cos(\phi(Y_m))$	MAIN	CBODY
CPHI(M)	$c_\phi(Y_m)$ or $\frac{\partial c}{\partial Y}(Y_m)$	MAIN	BLANK
CPHIY(J)	temporary storage for c_ϕ, (1)	EVAL, EVALSY, EVALPR	BLKØ1

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
CU(I,N,M); N≠1,NC	$U(X_n,Y_m)$, (2)	MAIN	BLANK
CU(I,1,M)	P at wall (I = 1); s at wall (I = 2); V_2 at wall (I = 3)	MAIN	BLANK
CU(I,NC,M)	c (I=1), c_z (I=2), c_ϕ (I=3)	MAIN	BLANK
CUP(I,N,M); N≠1,NC	$U^*(X_n,Y_m)$, (2) and (4)	MAIN	BLANK
CUP(I,1,M)	P^* at wall (I = 1); s^* at wall (I = 2); V_2^* at wall (I = 3), (4)	MAIN	BLANK
CUP(I,NC,M)	c^* (I=1), c_z^* (I=2), c_ϕ^* (I=3), (4)	MAIN	BLANK
CZ(M)	$c_z(Y_m)$	DECODE	BLANK
CZY(J)	temporary storage for c_z, (1)	EVAL, EVALSY, EVALPR	BLKØ1
D(N,M)	$\rho(X_n,Y_m)$	EVAL	BLANK
DCPHZ	$\frac{\partial c_\phi}{\partial Z}$	SHOCK	CSHOCK
DCZ	$\frac{\partial c}{\partial Z}$	SHOCK	CSHOCK
DCZZ	$\frac{\partial c_z}{\partial Z}$	SHOCK	CSHOCK
DDX	$1/\Delta X$	MAIN	BLKØ4
DDY	$1/\Delta Y$	MAIN	BLKØ4
DELII	parameter used for bent cone calcs.	INPUT	CBENT
DELTA	parameter used for bent cone calcs.	BODY	CBENT
DELZ	intercept value of cone	BODY, BODYR	CBODY
DINF	ρ_∞	INPUT	BLANK
DINF2	ρ_∞^2	MAIN	CDECODE

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
DINX	$V_\infty \cos\beta \cos\alpha \sqrt{\rho_\infty/p_\infty}$	MAIN	CSHOCK
DST	parameter used for bent cone calcs.	BODYR	CBENT
DY	ΔY	MAIN	BLKØ2
DYD3	$\Delta Y/3$	MAIN	CINTEG
DZ	ΔZ	MAIN	CBODYP
D1INF	$V_\infty \cos\beta \sin\alpha \sqrt{\rho_\infty/p_\infty}$	MAIN	CSHOCK
D2INF	$V_\infty \sin\beta \sqrt{\rho_\infty/p_\infty}$	MAIN	CSHOCK
ELIM	error limit for real gas iterative procedures	INPUT	BLKØ4
EPSQ	parameter used for bent cone calcs.	BODY	CBENT
FA	F_a	INTEG	CINTEG
FAZ	$\frac{\partial F_a}{\partial z}$	INTEG	CINTEG
FN	F_n	INTEG	CINTEG
FNZ	$\frac{\partial F_n}{\partial z}$	INTEG	CINTEG
FY	F_y	INTEG	CINTEG
FYZ	$\frac{\partial F_y}{\partial z}$	INTEG	CINTEG
GAMMA	Γ, (5)	INPUT, DECODE	BLKØ4
GA2	$2\Gamma_\infty/(\Gamma_\infty - 1)$, (5)	MAIN, JUMP	BLKØ4
GB	$1/(\Gamma_\infty - 1)$	MAIN	BLKØ4
GC(N,M)	parameter used in real gas iterative procedures	DECODE	CDECODE

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
GD	$(\Gamma - 1)/2$, (5)	MAIN, DECODE	BLKØ4
GE	$(\Gamma + 1)/2$, (5)	MAIN, DECODE	BLKØ4
GFF	$\Gamma + 1$, (5)	MAIN, DECODE	CDECODE
GG	$\Gamma^2 - 1$, (5)	MAIN, DECODE	CDECODE
GM(N,M)	Γ	MAIN, DECODE	BLKØ4
GIM1	$1-\Gamma_\infty$	MAIN	CDECODE
GX	Γ or γ	RGAS	RGASS
GY(M)	$T_{g_4}(Y_m)$	MAIN	CINTEG
GYMDY	g_y at $Y = -\Delta Y$	TRANGD	CTRANG
GYYMDY	g_{yy} at $Y = -\Delta Y$	TRANGD	CTRANG
GY1PDY	g_y at $Y = 1 + \Delta Y$	TRANGD	CTRANG
GYY1PDY	g_{yy} at $Y = 1 + \Delta Y$	TRANGD	CTRANG
HINF	h_∞	MAIN	CDECODE
HN	parameter used for bent cone calcs.	BODYR	CBENT
HOT2	$2h_\infty + V_\infty^2$	MAIN	BLKØ4
HX	h	RGAS	RGASS
IBN	= 0, spherical nose = 1, bent sphere-cone nose	INPUT	CBENT
ICFL	step count used after expansion discontinuity for option to reduce step size	MAIN	BLKØ3
ICHECK	Flag indicating predictor or corrector for DECODE	MAIN	CDECODE

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
IDYAW	= 0 symmetric case ($\phi_0 = \pi$) = 1 nonsymmetric case ($\phi_0 = 2\pi$)	MAIN	BLANK
IERRPR	number of previous steps to be printed out after error termination	INPUT	CSAVE
IJMPKT(M)	step count used after expansion and compression discontinuities for X derivative cancellation option	WALL	BLKØ3
IJUMP(M)	flag used to indicate that a discontinuity in body slope has been sensed	BODYP, MAIN	BLKØ3
IJUMP1(M)	flag used to control options in WALL when body slope discontinuity has been encountered	JUMP, BODYP, WALL	BLKØ3
ISWMOD	flag used to select options for wall point calculation	INPUT	CWALL
ISWSMO	for $0 \leq M \leq$ ISWSMO, entropy at wall is defined by extrapolation	INPUT	BLKØ4
JCFL	= 1, 2, or 3; tells which of μ_1, μ_2, μ_3 determines stability cond.	EVAL	CEVAL
K	step count	MAIN	BLANK
KCFL	number of steps to reduce step size after an expansion discontinuity	INPUT	CWALL
KFAC	step size is reduced by ΔZ/KFAC after an expansion discontinuity	INPUT	CWALL
LCNT	Maximum number of real gas iterations	INPUT	BLKØ4
MA	MC-1	MAIN	CINTEG
MAS	= MC (IDYAW=0) = MC-1 (IDYAW=1)	MAIN	CSAVE

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
MC	M, the number of points in the Y direction	INPUT	BLANK
MCFL	value of m where stability condition is determined	EVAL	CEVAL
MCP	MC+1	MAIN	CTRANG
MCMAX	max. ϕ pts. in dimension statements	DATA Card	
MOD1	= 1, second order accuracy for wall points = 0, first order accuracy for wall points	INPUT, WALL	CWALL
MX (REAL)	M_n^c $(z_c = 0)$	INTEG	CINTEG
MXZ (REAL)	$\frac{\partial M_n^c}{\partial z}$ $(z_c = 0)$	INTEG	CINTEG
MY (REAL)	M_y^c $(z_c = 0)$	INTEG	CINTEG
MYZ (REAL)	$\frac{\partial M_y^c}{\partial z}$ $(z_c = 0)$	INTEG	CINTEG
MZ (REAL)	M_a^c $(z_c = 0)$	INTEG	CINTEG
MZZ (REAL)	$\frac{\partial M_a^c}{\partial z}$ $(z_c = 0)$	INTEG	CINTEG
NA	NC-1	MAIN	BLKØ4
NC	N, the number of points in the X direction	INPUT	BLANK
NCFL	value n where stability condition is determined	EVAL	CEVAL
NCMAX	max. r pts. in dimension statements	DATA Card	
NFIRST	flag used in RGAS	MAIN, RGAS	RGASS

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
NJMKTC	max. number of steps to modify X derivatives at wall after a compression discontinuity	INPUT, MAIN	CWALL
NJMPKT	max. number of steps to modify X derivatives at wall after an expansion discontinuity	INPUT, MAIN	CWALL
NGAS	flag to determine gas mixture in RGAS	INPUT	RGASS
NSGD	number of ϕ values to be read in	INPUT	CTRANG
NSFD	number of $\bar{x}$ values to be read in	INPUT	CTRANF
NTARGET	number of target points for printout to be read in	INPUT	COUT
NTEST	≥ 0 then perfect gas < 0 then real gas	INPUT	RGASS
P(N,M)	$p(X_n, Y_m)$	EVAL	BLANK
PDIF	p_∞/ρ_∞	MAIN	CSHOCK
PHI(M)	$\phi(Y_m)$	MAIN	CBODY
PHIO	ϕ_o ($= \pi$ or 2π)	INPUT	BLANK
PHI1J, PHI2J	ϕ interval to turn JUMP subroutine off	INPUT	CBODYP
PI	π	MAIN	BLANK
PID2	$\pi/2$	BODYR	CBENT
PINF	p_∞	INPUT	BLANK
PWY(J)	temporary storage for p at wall, (1)	EVAL, EVALSY	BLKØ1
PZ	$\frac{\partial p}{\partial Z}$ at wall	WALL	CWALL
PZCOR(M)	$\tilde{\rho}^k_{1,m}$	WALL	CWALL

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
R(N,M)	$r(X_n, Y_m)$	TRANF	BLANK
RAD	$\pi/180°$	MAIN	BLANK
RRX	gas constant	RGAS	RGASS
SFD(N)	$\bar{x} = f(X_n)$	TRANFD	CTRANF
SFXD(N)	f_X	TRANFD	CTRANF
SFXXD(N)	f_{XX}	TRANFD	CTRANF
SGD(M)	$\phi = \phi_o\ g(Y_m)$	TRANGD	CTRANG
SGYD(M)	g_Y	TRANGD	CTRANG
SGYYD(M)	g_{YY}	TRANGD	CTRANG
SINF	s_∞	MAIN	BLANK
SINPHI(M)	$\sin[\phi(Y_m)]$	MAIN	CBODY
SPDIF	$\sqrt{\rho_\infty / p_\infty}$	MAIN	CSHOCK
SW(M)	s at wall	DECODE	BLKØ4
SWY(J)	*temporary storage for s at wall, (1)*	EVAL, EVALSY	BLKØ1
SZ	$\frac{\partial s}{\partial Z}$ at wall, (6)	WALL	CWALL
TANBN	tan(THETABN)	BODYR	CBENT
TANCO	tan(CONE)	BODY, BODYR	CBODY
TARGETZ (100)	Z values for targeting printout	INPUT	COUT
TF4(N,L)	T_{f_4}, (3)	TRANF	BLKØ2
TF6(N,L)	T_{f_6}, (3)	TRANF	BLKØ2
TF7(N,L)	T_{f_7}, (3)	TRANF	BLKØ2

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
TG4(J)	g_Y	TRANG	BLKØ2
TG5(J)	$T_{g_5} = g_{YY}/g_Y$, (1)	TRANG	BLKØ2
TG6(J)	g_{YZ}/g_Y, (1)	TRANG	BLKØ2
THETA	$\bar{y} = \phi/\phi_o$	TRANG	BLKØ2
THETABN	bend angle in bent cone calcs.	INPUT	CBENT
TX	temperature	RGAS	RGASS
U(N,M)	$u(X_n,Y_m)$	DECODE	BLANK
UNOR(N,L)	(N = 1,2,3, only) A, (3)	EVAL	BLKØ1
UZCOR(I,M)	$F^k_{N,m} + F^k_{N-2,m} - 2F^k_{N-1,m}$	SHOCK	CSHOCK
V(N,M)	$v(X_n,Y_m)$	DECODE	BLANK
VINF	V_∞	MAIN	COUT
VOWY(J)	v/w at wall, (1)	EVAL, EVALSY	BLKØ1
V1INF	$V_\infty \sin\alpha \cos\beta$	MAIN	CDECODE
V2(J)	V_2 at wall, (1)	EVAL, EVALSY	BLKØ1
V2INF	$V_\infty \sin\beta$	MAIN	CDECODE
V2Z	$\frac{\partial V_2}{\partial Z}$ at wall	WALL	CWALL
W(N,M)	$w(X_n,Y_m)$	DECODE	BLANK
WINX	$V_\infty \cos\alpha \cos\beta$	MAIN	CDECODE
X(N)	X_n	MAIN	BLKØ2
XINDEF	Undefined quantity		COUT

FORTRAN Symbol	Description (in Part I notation)	Principal Defining Routine	Common Block
XPHI(N,L)	X_ϕ, (3)	TRANF	BLKØ2
XR(N,L)	X_r, (3)	TRANF	BLKØ2
XZ(N,L)	X_z, (3)	TRANF	BLKØ2
X1KINEQ	Θ	MAIN	CDECODE
X1KINP2	$\Theta + 2$	MAIN	CDECODE
Y(M)	Y_m	MAIN	BLKØ2
YAW	β	INPUT	COUT
YPHI(J)	Y_ϕ, (1)	TRANG	BLANK
YZ(J)	Y_z, (1)	TRANG	BLANK
Z	Z^{k+1} or Z^k	MAIN	CBODY
ZBB(M)	parameters used in bent cone calcs.	MAIN, BODY	CBENT
ZEND	z_{end}, final z value	INPUT	COUT
ZMAXS	parameter used in bent cone calcs.	BODY, BODYR	CBENT

Remarks:

(1) These quantities are not fully stored. The J index identifies a line Y = constant and is either 1, 2, or 3. For quantities indexed as (N,J), the index N refers to X = X_n.

(2) Quantities indexed (I,N,M), (I,N,J), or (I,N,L) are 4 dimensional column vectors. The index I = 1,2,3, or 4 indicates the component (from top to bottom). The exceptions are CUP(I,1,M), CU(I,1,M), CUP(I,NC,M), and (CUI,NC,M).

(3) These quantities are not fully stored. The L index identifies a line Y = constant and is either 1 or 2. For quantities indexed as (N,L), the index N refers to X = X_n.

(4) At certain points in the MAIN, these locations store the numerical Z-derivatives of the indicated quentities for the predictor step.

(5) In real gas calculations, these quantities are defined with $\Gamma = \Gamma_\infty$ in MAIN. In perfect gas calculations, these quantities are constants.

9. MAIN

MAIN is divided into two sections. Section 1 is comprised of all operations performed at the initial entry to the program; hence, this section is executed only once in the entire calculation. Section 2 contains the predictor-corrector marching scheme and is, therefore, executed repeatedly. Each cycle through this section corresponds to one marching step of the calculation. The basic operation of MAIN, and hence the entire program, is shown schematically in Fig. 8. Each rectangle in the figure represents a major subsection of the program. Note well, these functional rectangles can be easily identified in the listing by locating the corresponding comment cards. The individual subsections are described in Secs. 9.1 and 9.2 below.

9.1 Section 1

Input - The initial flow field data is input from tape and rezoned if necessary. Also, various program controls and parameters are input from cards. Specific instructions and descriptions of both of these inputs are given in the User's Manual.

Initializations and Parameters - Various fixed parameters used throughout the calculation are computed and the X,Y,CU, and ASQ and GM arrays are initialized (except for N=1 and NC) using the initial flow field data.

Preliminary Predictor Loop - This loop computes the derivatives $\frac{\partial c}{\partial Z}$, $\frac{\partial c_\phi}{\partial Z}$ $\frac{\partial c_z}{\partial Z}$, $\frac{\partial U}{\partial Z}$, $\frac{\partial P}{\partial Z}$, $\frac{\partial V_2}{\partial Z}$, $\frac{\partial s}{\partial Z}$ which are required in the predictor for the first step. These derivatives are determined using (3.9a) - (3.9c), (3.6a), (3.16) - (3.18); their values are stored in the CUP and CP arrays.

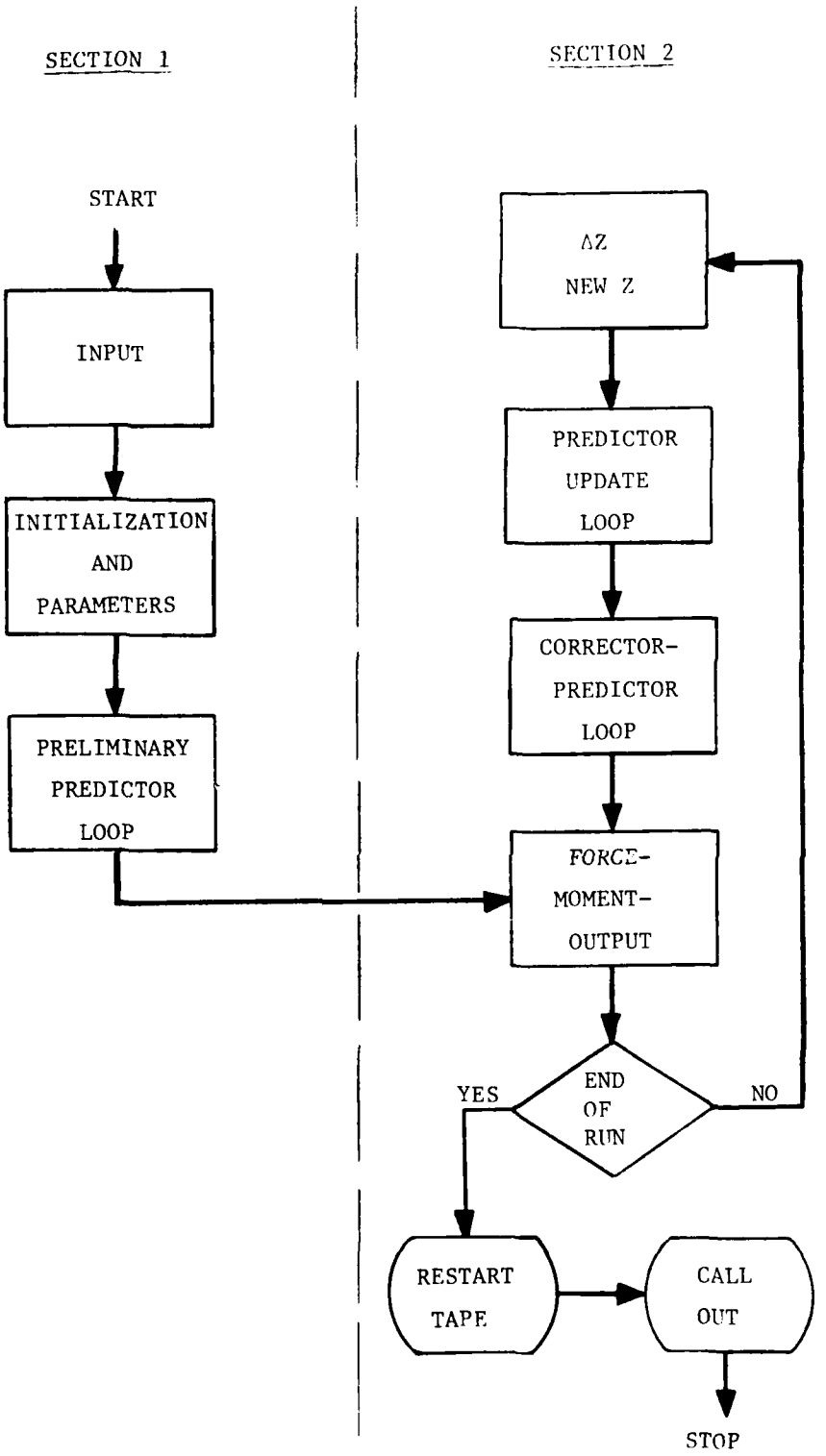


Fig. 8. Flow chart of MAIN

Here the CU, ASQ, and GM arrays (see sec. 8) for N=1 and NC are initialized using wall and shock quantities from the initial data. In each pass through this loop, the above operations are performed on the adjacent lines M-1 and M*. The quantities K1,K2 (= 1 or 2) are indices which identify certain quantities determined in the previous pass through the loop (see the comment statements in the listings for more details). When the loop has been completed, the stability parameter, CFL, for calculating the step size for the first step has been determined (in EVAL).

9.2 Section 2

ΔZ,new Z - In this subsection, the step size ΔZ is determined using the value of the stability parameter, CFL, obtained in the previous cycle or, in the case of the first step, in the Preliminary Predictor Loop. After an expansion discontinuity, ΔZ can be reduced if this option is selected by the user. The new value of Z is then obtained by incrementing the previous value of Z by ΔZ. This subsection also contains tests to determine if an axes shift is necessary in bent cone calculations (see User's Manual for details).

Predictor Update Loop - In this loop the predicted values of the conservation vector U, the wall quantities P, V_2, s, and the shock geometry c, c_z, c_ϕ for the new value of Z are determined using (3.6a), (3.13) , and (3.10). Recall that the Z-derivatives appearing in these equations were computed in the previous cycle or, in the case of the first step, in the Preliminary Predictor Loop and were stored in the CUP and CP arrays. In the execution of the loop, the predicted values, as they are determined, replace the derivative values in these arrays.

*The set of computational points {X(N),Y(M)} where M is fixed and N=1,2,···,NC will be referred to as the line M.

Corrector-Predictor Loop - This loop contains the major part of the marching step calculation. In the loop, various operations are performed on the adjacent lines* M, M-1, and M-2. The quantities K1,K2 (= 1, or 2) and J1,J2,J3 (= 1, 2, or 3) are indices which identify certain quantities determined in previous passes through the loop (see, the comment statements in the listing for more details). In each pass through the loop, the following operations are performed in the sequence indicated**:

(i.) On the line M, the predicted values of the flow variables p,ρ,w,u,v,a^2 are determined from the predicted values of U,P,V_2S,c_ϕ, c_z, c (c.f., DECODE, see Sec. 10.2). Note that the predicted values of U,P,V_2,s,c_ϕ,c_z, c were determined in the Predictor Update Loop.

(ii.) The corrected value of c at Y = Y(M) is determined using (3.11).

(iii.) On the line M-1, the corrected values of the conservation vector U, the wall quantities P,V_2,s, and the shock geometry c_ϕ,c_z are determined using (3.6b), (3.14), (3.11). These are stored in the CU array.

(iv.) On the line M-1, the corrected values of p,ρ,u,v,w,a^2 are determined using the quantities obtained in (iii), and the corrected value of c determined in (ii.) for the previous pass through the loop; c.f., DECODE, Sec. 10.2. Note that the corrected values obtained here replace the

*Ibid

**The sequence of operations indicated here is necessarily modified for the first and last two passes through the loop (see the listing for details).

predicted values in the P,D,U,V,W,ASQ,CPHI,CZ arrays on the line M-1. The predicted values on M-1 are no longer needed in the calculation.

(v.) If for Y = Y(M-1), a discontinuity in body slopes has been found (by BODYP), the true body geometry is substituted for the modified geometry (in BODYPP) and discontinuities in the surface flow variables are computed (in JUMP). This procedure is discussed in detail in Sec. 4.1.

(vi.) On the line M-1, the local stability parameters $\mu/(w^2 - a^2)$ (c.f., Sec. 3.6) are computed using the corrected values of the flow variables and the maximum of these parameters taken over all previously computed lines is up-dated to include the line M-1 (in EVAL). Note that the stability parameters being considered here are for the next Z step.

(vii.) On the line M-2, the derivatives $\frac{\partial U}{\partial Z}$, $\frac{\partial P}{\partial Z}$, $\frac{\partial V_2}{\partial Z}$, $\frac{\partial s}{\partial Z}$, $\frac{\partial c}{\partial Z}$, $\frac{\partial c_z}{\partial Z}$, $\frac{\partial c_\phi}{\partial Z}$ required for the predictor for the next step are computed using (3.6a), (3.16)-(3.18), and (3.9a)-(3.9c). These quantities are stored in the CUP and CP arrays.

Note that after the Predictor-Corrector Loop has been completed, the arrays CU,C,CZ, CPHI,P,U,V,W,ASQ contain corrected values of the corresponding quantities at the new value of Z. The arrays CUP and CP contain the values of the Z derivatives required for the predictor step of the next cycle. Furthermore, the stability parameter, CFL, for determining the size of the next step has been determined.

Force and Moments, Output - On this subsection, the aerodynamic force and moment results are computed (in INTEG) and outputs are performed. The flow variables and the force and moment results for the current step

are output to a binary tape (TAPE 16). Also, if selected by the user, the current flow field results are output on-line (in FIELD). The various options available for on-line printout are discussed in the User's Manual.

If the calculation is to be continued (i.e., the current value of Z is less than ZEND and the step count K is less than the maximum number of steps selected by the user), the control is returned to the beginning of Section 2 and the cycle for the next step is performed. If, on the other hand, the calculation is to be terminated, the current flow variables and the force and moment quantities are output to a binary tape (TAPE 17). This tape serves as an input tape if the calculation is to be restarted from the current station in a different run (see the User's Manual for instructions on restarting). The final operation before termination is the on-line printout of the surface pressure data and the force and moment coefficients (see OUT, Sec. 12.3).

10. SUBROUTINES USED IN THE FLOW FIELD CALCULATION

The subroutines discussed in this section are all called from MAIN in the Preliminary Predictor Loop and the Corrector-Predictor Loop. These two loops and the subroutines of this section constitute the complete algorithm for computing one marching step of the flow field calculation. All the subroutines in this section are line operations; i.e., when the subroutine operation depends on X and Y, all points X(N), Y(M) where M is fixed and N = 1,2,···,NC are considered for each entry. When the subroutine operation depends exclusively on Y, only the value Y = Y(M) is considered for each entry. Generally, the inputs to and outputs from these subroutines are via the COMMON BLOCK (see Sec. 8). Thus the arguments in the calling sequences of these subroutines only contain indices which identify the input/output quantities on the particular line for which the subroutines are to operate (the exceptions to this are DECODE (see, Sec. 10.2) and JUMP (see, Sec. 10.4).

10.1 BODYP, ENTRY BODYPP

Calling sequence: Call BODYP(M,J3)

where	M	is	the line index; i.e., Y = Y(M).
	J3 (= 1,2, or 3)	is	the identification index corresponding to the line M for the body parameters A1,A2,··,A5,A7.

Description: This routine defines the body parameters A1(J3),···, A5(J3),A7(J3) for the line M (for definitions of these quantities, see Sec. 8). These quantities are computed using the body geometry contained in B(M), BPHI(M), BZ(M), BZZ, BZPHI, and BPHPHI (determined in BODY). Note that BODYP is always called after BODY in the MAIN.

This routine also tests for discontinuities in body slope (see Sec. 4.1 for a discussion of the procedures used when such discontinuities are present). If a discontinuity is found and if subroutine JUMP is to be used, the flag IJUMP(M) is set to 1 and the old (previous) values of b_z, b_ϕ, $b_{z\phi}$, and b_{zz} (stored in BZO(M), BPHIO(M), etc.) are used to define A1(J3), A2(J3), etc. In this case, the true values of the body derivatives are stored temporarily in BZT(J3), BPHIT(J3), etc. The entry point BODYPP is used only when the JUMP routine is used. It is called from the Corrector-Predictor Loop for the purpose of redefining the arrays A1(J3), A2(J3), etc. using the true values of the body derivatives.

10.2 DECODE

Calling sequence: CALL DECODE (M,CV, J,NDIM,MDIM)

where

M	is	the line index; i.e., Y = Y(M)
CV	is	either the CU or CUP array
J (= 1,2, or 3)	is	the identification index corresponding to the line M for the body parameter A2
NDIM	is	the value of the dimension for the N index of CV
MDIM	is	the value of the dimension for the M index of CV

Description: This routine defines the flow variable arrays P,D,U,V,W,ASQ and the shock arrays C,CZ,CPHI for the line M. The flow quantities at the wall (N = 1) are determined from the values of P, V_2, and s contained in the CV array for N = 1 using p = exp(P), (2.4b), and (3.15). For interior points 1 < N < NC, the flow quantities are determined from the values of the

conservation vector contained in the CV array using the procedure described in Sec. 3.2. At the shock N = NC, the flow variables are determined using the Rankine-Hugoniot relations with c, c_ϕ, c_z contained in the CV(1,NC,M) CV(3,NC,M),CV(2,NC,M) arrays (see Sec. 3.3). The thermodynamic properties needed in these procedures are supplied from subroutines RGAS and HGAS.

This routine also contains the selective smoothing procedure given in section 4.1. When an interior point has a negative pressure, the conservation vector CV is redefined at that point using (4.11) and flow variables are recomputed.

10.3 EVAL, ENTRY EVALSY, ENTRY EVALPR

Calling sequence: CALL EVAL (L,M,IT,JSG,JCG,JCFCE)

where

L	is	0 in the predictor and 1 in the corrector
M	is	the line index; i.e., Y = Y(M)
IT (= 1 or 2)	is	the identification index corresponding to the line M for XZ,XR,XPHI,TF4, TF6,TF7
JSG (= 1,2, or 3)	is	the identification index corresponding to the line M for YPHI,YZ,TG4,TG5,TG6
JCG	is	the identification index corresponding to the line M for CG,VOWY,PWY,SWY,V2
JCFCE (= 1 or 2)	is	the identification index corresponding to the line M for CF,CE,UNOR

Description: This routine determines the flux vectors F and G and the source term E on the line M using the definitions given in (3.3a), (3.3b), and (3.3c). The flow variables p, ρ, etc. used in these equations

are the current values contained in the COMMON arrays P,D, etc. on the line M. On this line, these arrays may contain predicted values (when L = 1) or corrected values (when L = 0). The other quantities required in (3.3a) - (3.3c) are XZ,XR,XPHI,TF4,TF6,TF7 (from COMMON using the index IT) and YZ,TG4,TG5,TG6 (from COMMON using the index JSG). The values of F and E that are obtained are stored in COMMON arrays F and E using the JCFCE index; the value of G is stored in the CG array using the index JCG. This routine also computes and/or stores various quantities used in WALL and SHOCK. These are A for N = 1,2,3 (stored in the COMMON array UNOR using the index JCFCE), the wall values (N = 1) of v/w,p,s,V_2 (stored in the COMMON arrays VOWY,PWY,SWY, and V2, respectively, using the index JCG) and the shock slopes c_ϕ,c_z (stored in COMMON arrays CPHIY and CZY using the index JCG).

For each step Z, of the calculation, EVAL is entered twice; once when L = 0 (predictor) and once when L = 1 (corrector). When this routine is entered with L = 0 (corrected values in P,D, etc.) the local stability parameter $\mu/(w^2 - a^2)$ (c.f., Sec. 3.6) is computed and compared to the maximum of these quantities taken over the previously executed lines. Note that the value of CFL in COMMON after EVAL has been executed is the maximum of the local values taken over the lines 1,2,$\cdots$,M.

The entry EVALSY is used in the symmetric problem to define the COMMON arrays CG,VOWY,PWY,V2,CPHIY, and CZY on the "fringe" lines corresponding to $-\Delta Y$ and $1 + \Delta Y$. The flow variables on these planes are determined using the symmetry conditions (3.21) and (3.22). For this entry, the argument M is 2 (for $Y = -\Delta Y$) and MC-1 (for $Y = 1 + \Delta Y$); the index IT identifies the

elements of the COMMON array TG5 corresponding to Y = 0 or Y = 1; and the index JSG identifies the elements of the arrays YZ and YPHI corresponding to Y = - ΔY or Y = 1 + ΔY.

The entry EVALPR is used in the non-symmetric problem to define the COMMON arrays CG,VOWY,PWY,VZ,CPHIY, and CZY on the planes -ΔY and 1 using (3.24). In this entry M is MC-1 for the plane Y = - ΔY and 1 for the plane Y = 1; the index JCG identifies the elements of the COMMON arrays YZ and YPHI on these planes.

10.4 JUMP

Calling sequence: CALL JUMP (DBP,DBZ,MB)

where

DBP is $[(b_\phi/b)_- - (b_\phi/b)_+]$ (see Part I notation)

DBZ is $[(b_z)_- - (b_z)_+]$ (see Part I notation)

MB is the line index; i.e., Y = Y(MB)

Description: This routine computes the discontinuities in the flow variables p,ρ,u,v,w,a^2 at the wall (N = 1) associated with discontinuities in b_z and/or b_ϕ; c.f., Sec. 4.1. This routine is called only when a discontinuity is found (in BODYP) for Y = Y(MB) (i.e., the flag IJUMP(MB) = 1). The routine computes the surface flow variables on the downstream side of the discontinuity (subscripted + in Sec. 4.1) using the formulas given in Sec. 4.1. The flow variables with subscript - in Sec. 4.1 are input from COMMON in the P,D, etc. arrays with N = 1. The output flow quantities (corresponding to the subscript + in Sec. 4.1) are stored in these locations when the routine is executed. The routine also sets the flag IJUMP1(MB) (see comment cards of JUMP for details), puts ISWSMO = 0, if there is a compression corner, and starts the counts ICFL and IJMPKT(MB). These are used in WALL and the MAIN to control

various special procedures for the wall point calculations downstream of the discontinuity (see, Sec. 4.1 for details).

10.5 TRANF, ENTRY TRANFW

Calling sequence: CALL TRANF(M,J,I)

where	M	is	the line index; i.e., Y = Y(M)
	J (= 1,2, or 3)	is	the identification index corresponding to the line M for YZ,YPHI,TG6
	I (= 1 or 2)	is	the identification index corresponding to the line M for XR,XZ,XPHI,TF4, TF6,TF7

Description: This routine defines the arrays R,XR,XZ,XPHI,TF4,TF6, and TF7 on the line M using the definitions given by (3.3g), (3.3i), and $r = b + \bar{x}(c - b)$. The quantities $\bar{x} = f$, f_Z, f_Y, etc. appearing in these equations must be specified in this routine by the user. Specifically, defining relations for the following FORTRAN variables must be programmed into this routine as functions of (X,Y,Z):

$$SX = f \ , \quad SFX = f_X \ , \quad SFY = f_Y \ , \quad SFZ = f_Z$$

$$SFXX = f_{XX} \ , \quad SFYX = f_{YX} \ , \quad SFZX = f_{ZX}$$

(see Sec. 4.3 for a discussion of the requirements on the choice of the mapping function f(X,Y,Z) and a non-trivial example). In the routine, any of the above variables which do not depend explicitly on X can be defined outside the loop on N; all variables defined which depend on X must be defined inside the loop on N (see listing). Note that version 1 of this routine given in Appendix D has two options. One is for the case of no clustering in the radial direction; i.e., f(X,Y,Z) = X, hence, SX = f = X(N), SFX = 1.0, and SFY = SFZ = SFXX = SFYX = SFZX = 0. The other option allows the user to select the desired mesh spacing in the radial direction by directly inputing the values of $\bar{x} = f$ to be used in the calculation (see sec. 4.3 for details). When this option is used, the necessary data is read-in and the quantities f, f_X, and f_{XX} (all Y and Z derivatives of f are zero) are computed

in TRANFD (see sec. 12.5). These derivatives are input to TRANF via the COMMON arrays SFD, SFXD, and SFXXD. The other quantities needed in the evaluations of R,XR, etc. are YZ,YPHI,TG6 (input from COMMON using the index J) and B,BZ,BPHI,C,CZ, and CPHI (input from COMMON using the index M). The quantities XZ,XPHI,TF4,TF6, and TF7 are stored in COMMON using the index I. Note, this routine is called only once per line. For this call, C contains the corrected value, CPHI and CZ contain predicted values. When the corrected values of CZ and CPHI are determined, the quantities XZ,XPHI,TF6,TF7 (the only ones depending on CZ and CPHI) are updated in the Corrector-Predictor Loop.

The entry point TRANFW is used only once in the program (called from Section 1 of MAIN). Its purpose is to print out on the heading page an identification of the particular mapping function f used in the routine.

10.6 TRANG, ENTRY TRANGW

Calling sequence: CALL TRANG(YY,M,J)

where

YY	is	the value of Y = Y(M)
M	is	the line index
J	is	the identification index corresponding to the line M (where M is such that YY = Y(M)) for YPHI,YZ,TG4,TG5,TG6

Description: This routine defines the quantities THETA,YPHI,YZ,TG4,TG5, TG6 for Y = YY

where

$$\text{THETA} = \bar{y} = g \ , \quad \text{YPHI} = 1/(\phi_o g_Y) \ , \quad \text{YZ} = -\, g_Z/g_Y$$

$$\text{TG4} = g_Y \ , \quad \text{TG5} = g_{YY}/g_Y \ , \quad \text{TG6} = g_{ZY}/g_Y \ .$$

These quantities are stored in COMMON using the J index. The quantities g, g_Y, g_Z, etc. appearing in the above definitions must be specified in this routine by the user. Specifically, defining relations for the following FORTRAN variables must be programmed into this routine as functions of (YY,Z):

$$\text{SG} = g \ , \quad \text{SGY} = g_Y \ , \quad \text{SGZ} = g_Z$$

$$\text{SGYY} = g_{YY} \ , \quad \text{SGYZ} = g_{YZ}$$

(see Sec. 4.3 for a discussion of the requirements on the choice of the mapping function g(Y,Z) and a non-trivial example). Note that version 1 of this routine given in Appendix D has two options. One is for the case of no clustering in the azimuthal direction; i.e., g(YY,Z) = YY, hence, SG = YY, SGY = 1.0 SGZ = 0, SGYY = 0, SGYZ = 0. The other option allows the user to select the desired mesh spacing in the ϕ-direction by directly inputing the values of ϕ(M) to be used in the calculation (c.f., sec. 4.3). For this option, the necessary data is read-in and the quantities g, g_Y, g_{YY} (Z derivatives of g are zero) are computed in TRANFD (see sec. 12,6). These are input to TRANG via the COMMON arrays SGD,SGYD,SGYDD using the index M.

The entry point TRANGW is used only once in the program (called from Section 1 of MAIN). Its purpose is to print out on the heading page an identification of the particular mapping function g used in the routine.

10.7 WALL

Calling sequence: CALL WALL(M,JR,JL,JSG,IF,L)

where

M	is	the line index; i.e., Y = Y(M)
JR,JL (= 1,2, or 3)	are	line identification indices used for the Y differences (i.e., corresponding to M and M + 1, respectively, for the predictor and to M - 1 and M, respectively, for the corrector)
JSG (= 1,2, or 3)	is	the identification index corresponding to the line M for YPHI,YZ,TG5,TG6, A3,A4,A5,A7
IF (= 1 or 2)	is	the identification index corresponding to the line M for XR,TF6,TF7,UNOR
L	is	0 in the predictor and 1 in the corrector

Description: In this routine, the derivatives $\frac{\partial P}{\partial Z} = \frac{1}{p}\frac{\partial p}{\partial Z}$, $\frac{\partial V_2}{\partial Z}$, and $\frac{\partial s}{\partial Z}$ are computed for use in both the predictor and corrector for the wall

points (X = 0, N = 1); c.f., Sec. 3.4. This routine contains both the formulations described in Section 3.4. These formulations are denoted in the code using the following terminology:

MOD 0 indicates that (3.16a) and (3.17a) are used in (3.16) and (3.17), respectively.

MOD 3 indicates that (3.16b) and (3.17b) are used in (3.16) and (3.17), respectively.

When the flag ISWMOD = 0, MOD 0 is used; when the flag ISWMOD = 3, MOD 3 is used. This routine also contains the option for using second order accurate differencing for the wall points; i.e., (3.19). This option can be used with either the MOD 0 or the MOD 3 formulations. It is controlled by the MOD1 flag; i.e., second order accuracy is used when MOD1 = 1 and is not used when MOD1 = 0. Another option contained in this routine is that of wall entropy reduction (see, Sec. 4.2). This option can be used with any combination of the other options; it is controlled by the flag ISWSMO. When ISWSMO $\neq$ 0 and M $\leq$ ISWSMO, the routine computes the wall value of s (not its derivative) using the extrapolation formula (4.10.2).

Initially, the user can select which of the above options are to be used (see User's Manual for instructions). When discontinuities in body slope are encountered on the line M, modifications in the computational procedure at the wall are automatically made on the line M and other options come into play (see Sec. 4.1 for a discussion of these procedures). The wall point calculation on the line M is controlled by the flag IJUMP1(M). When IJUMP1(M) = 0 the user selected options are used; i.e., there is no body slope discontinuity on the line M. Immediately after a discontinuity is found on the line M by BODYP, IJUMP1(M) (in JUMP or BODYP) is set to: 2 if JUMP finds no pressure change across the discontinuity

(or if JUMP is not used), 3 if JUMP finds a pressure change due to an expansion, 4 if JUMP finds a pressure change due to a compression. When IJUMP1(M) = 2, the wall point calculation is as follows for the remainder of the run:

(i.) MOD 0 is used on the line M

(ii.) the entropy reduction option is turned off (ISWSMO is set to zero in JUMP) if a compression corner exists

(iii.) on the line M, second order accuracy is turned off (if originally selected)

When IJUMP1(M) = 3 or 4, (i.) - (iii.) are used with the option for zeroing the X derivative terms in (3.16) (c.f., Sec. 4.1). This option will then be used for ensuing marching steps on the line M until the test (4.10.2) is satisfied on the line M or until a maximum number of steps downstream from the discontinuity have been taken. The maximum number of steps used in this procedure can be chosen by the user and can differ for expansions or compressions. When either of the above criteria are satisfied, the flag IJUMP1(M) is set to 2 for the remainder of the run. Note, the associated counting and testing is performed in this routine.

In the evaluation of $\frac{\partial P}{\partial Z}$, $\frac{\partial V_2}{\partial Z}$, $\frac{\partial s}{\partial Z}$, the quantities which must be Y differenced are input from COMMON in the arrays PWY,VOWY,SWY,V2 or CG (N = 1) depending on whether the MOD 0 or the MOD 3 formulation is used. The Y differences (forward for the predictor, backward for the corrector) are controlled by the MAIN using the indices JR,JL. The final results for $\frac{\partial P}{\partial Z}$, $\frac{\partial V_2}{\partial Z}$, and $\frac{\partial s}{\partial Z}$ are returned to MAIN using the COMMON variables PZ,V2Z, and SZ, respectively; however, when the entropy is extrapolated the value of s, not its derivative, is returned in SZ.

10.8 SHOCK

Calling sequence: CALL SHOCK (M,JR,JL,JSG,IF,L)

where

M	is	the line index, i.e., Y = Y(M)
JR,JL (=1,2, or 3)	are	line identification indices used for for Y differences (i.e., corresponding to M and M-1, respectively, for the predictor and to M-1 and M, respectively, for the corrector)
JSG (=1,2, or 3)	is	the identification index corresponding to the line M for YPHI, and YZ
IF (=1 or 2)	is	the identification index corresponding to the line M for CF, CG, and CE
L	is	0 in the predictor and 1 in the corrector

Description: In this routine, the derivatives $\frac{\partial c}{\partial Z}$, $\frac{\partial c_\phi}{\partial Z}$, and $\frac{\partial c_z}{\partial Z}$ are computed for use in both the predictor and corrector steps; c.f., Sec. 3.3. The quantities to be Y differenced are input from COMMON in the arrays CZY, CPHIY, and CG. The Y differences (forward for the predictor, backward for the corrector) are controlled by MAIN using the indices JR,JL. The final results for $\frac{\partial c}{\partial Z}$, $\frac{\partial c_\phi}{\partial Z}$, $\frac{\partial c_z}{\partial Z}$ are returned to MAIN using the COMMON variables DCZ,DCPHZ,DCZZ, respectively.

11. AUXILIARY SUBROUTINES

11.1 INTEG

Calling sequence: CALL INTEG (IFLAG)

where IFLAG is 0 for the first entry and 1 for all other entries

Description: This routine numerically integrates the surface pressure results to obtain the components of the aerodynamic force and moment and their z derivatives. The definitions of these quantities and the procedures used for their evaluation are described in Section 5. In this routine, the center for the moment is the origin; i.e., $z_c = 0$. When the routine is called for the initial value of Z ($= z_o$), IFLAG = 0 and only the z derivatives are computed. When the routine is called for each subsequent step, Z^k, IFLAG = 1 and the z derivatives and the components are computed. The latter corresponding to the body truncated at $z = Z^k$. The quantities required for evaluating the integrands in (5.1) - (5.5) are input using the COMMON arrays P(N = 1),B,BZ,BPHI,GY,COSPHI,SINPHI. The results are stored in the COMMON arrays FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ, MYZ,MZZ. Note that (for the symmetric problem only the non-zero quantities are computed. Also, the numerical formulas for determining the z derivatives are slightly different from those used for the non-symmetric problem.

11.2 INTRPL

Calling sequence: CALL INTRPL (L,X,Y,N,XX,YY)

where

L	is	the number of pts. in the input arrays X and Y
X	is	the array containing abscissas of the input table to be interpolated
Y	is	the array containing the ordinates of the input table to be interpolated
N	is	the dimension of the XX and YY arrays
XX	is	the array containing the abscissas at which the interpolant is to be evaluated
YY	is	the array containing the ordinates obtained by evaluating the interpolant at the XX values

Description: This routine is called from REZONE and SHFAX for the purpose of interpolating the given points (X,Y) to find the values YY corresponding to the specified XX values. The routine given in Appendix D uses standard linear interpolation. The routine assumes that the input data in the X and XX arrays are increasing; i.e., X(I) < X(I + 1) and XX(I) < XX(I + 1).

11.3 REZONE

Calling sequence: CALL REZONE (NCNEW,MCNEW,ROLD,PHIOLD,DCUB,DARR1,DARR2, DARR3,DARR4,ND1M,MD1M)

where

NCNEW	is	the number of points in X direction for the run
MCNEW	is	the number of points in Y direction for the run
ROLD	is	a dummy array used to store R array from input tape
PHIOLD	is	a dummy array used to store the PHI array from input tape

DCUB	is	an array used to store the P,U,V,W arrays from input tape*
DARR1,DARR2,DARR3	are	temporary storage arrays corresponding to the index M
DARR4	is	temporary storage array corresponding to the index N
NDIM	is	the value of the dimension for the N index
NDIM	is	the value of the dimension for the M index

Description: This routine must be used when the coordinates r and ϕ of the initial data on the input tape are different than the r,ϕ coordinates corresponding to the computational mesh for the run. The routine generates initial data at the points of the computational mesh by interpolating the data obtained from the input tape. The interpolations are performed in INTRPL (see, Sec. 11.2). Instructions for using the routine are given in the User's Manual.

11.4 RGAS

Calling sequence: CALL RGAS (PX, RX, SX, NUMX)

where

PX	is	pressure
RX	is	density
SX	is	entropy
NUMX	is	a flag indicating mode of operation (see below)

Description: This routine was developed at NASA Ames to provide thermodynamic properties of 13 different gas mixtures. The variable NGAS indicates which mixture is to be used. When NUMX is 4, pressure

*Note that the use of DCUB in this routine requires that the arrays P,U,V,W are consecutive in COMMON.

and density are input and entropy (SX), enthalpy (HX), sound speed (AX) and temperature (TX) are found directly using table look-ups. When NUMX is 5, entropy and pressure are input and an iterative procedure is used to determine density. This value of density and the given pressure are then used to find the other variables (as in the case NUMX=4). If NTEST is non-negative, perfect gas relations are used. In this mode GX contains γ and RRX the perfect gas constant. The variables AX,HX,TX,RRX,GX,NTEST,NGAS are transmitted to and from the routine via COMMON/RGASS. For a more detailed description of RGAS see the following: Eaton, R. R. and Larson, D. E. "Improved Real Gas Routines for Sandia's NASA Ames Flowfield Program", SAND 75-0493, Feb., 1976.

11.5 HRGAS,ENTRY ARGAS

Calling sequence: CALL HRGAS (PX,RX,QX,N1)

where

PX	is	pressure
RX	is	density
QX	is	sound speed square
N1	is	a flag indicating mode of operation

Description: This routine is a shortened version of RGAS which calculates only enthalpy (HX) and sound speed given the pressure and the density. This routine is called only in real gas calculations. If N1 = 2 only enthalpy is returned; for N1 = 1 both quantities are returned. ENTRY ARGAS is similar except that the values of pressure and density are those defined in the last HRGAS call. This subroutine must be used in conjunction with RGAS since this latter routine loads the COMMON arrays used by HRGAS.

11.6 SERCH, LOCATE

These routines are called only in RGAS and HRGAS. They have no direct use in the flow field calculation and therefore will not be discussed here.

11.7 SHFAX, SHFAXD

Calling sequence: CALL SHFAX (I,NDIM,MDIM,RN,UN,VN,WN,PN,DN,CPP, CZO,CON,CN,CNO,CPHIO)

where

I	is	(=1,2) a flag indicating which of the two criteria is used for axis shifting (c.f., sec. 11, User's Manual)
NDIM	is	the value of the dimension for the N index
MDIM	is	the value of the dimension for the M index
RN,UN,VN,WN,PN,CPP, CZO,CON,CN,CNO,CPHIO	are	dummy storage locations

Description: SHFAX is called when the coordinate system is to be shifted by a parallel displacement of the z axis in the x-z plane (see Fig. 1). This procedure is used in bent nose calculations. For a discussion of this mode of calculation see the user's manual. The coordinate system is shifted by the amount ZAS (see user's manual for explanation of this and other parameters used). The x,y,z coordinates of the new (shifted) origin in the original system is (- ZAS,0,0). The routine determines the flowfield variables and shock geometry for restarting the calculation on the initial plane in the shifted coordinates which corresponds to the last computational plane in the original coordinate system. This is performed using bilinear interpolation of the known flow field in the original coordinate system. The aerodynamic moments are also

referenced to the new origin. The flow variables, shock geometry, and moments in the original coordinates are input from COMMON; also the corresponding quantities in the shifted coordinates are output using COMMON. The dummy storages are used internally for the interpolations. SHFAX is called from SHFAXD (see below).

Calling sequence: CALL SHFAXD (I,NDIM,MDIM, CV,CVP,CP,CZO,CON, CN,CNO,CPHIO)

I	is	(=1,2) a flag indicating which of the two criteria is used for axis shifting (sec. 11, User's Manual)
NDIM	is	the value of the dimension for the N index
MDIM	is	the value of the dimension for the M index
CV,CVP,CP,CZO,CON,CN, CNO,CPHIO	are	dummy storage locations

Description: This routine calls SHFAX. It sets-up the dummy storages used in SHFAX.

12. INPUT-OUTPUT SUBROUTINES

12.1 BODY, ENTRY BODYW, ENTRY BODYR

Calling sequence: CALL BODY(M)

where M is the line index; i.e., Y = Y(M)

Description: This routine inputs to the program the values of the body shape function, $b(\phi,z)$, and certain of its derivatives (see below) for ϕ = PHI(M) and z = Z. Where, the values of PHI(M) and Z are input to the routine from COMMON. The coordinate system in which the function $b(\phi,z)$ is specified is illustrated in Fig. 1. The specific COMMON variables which are defined in this routine are:

$$B(M) = b \ , \quad BZ(M) = b_z \ , \quad BPHI(M) = b_\phi$$

$$BZPHI = b_{z\phi} \ , \quad BZZ = b_{zz} \ , \quad BPHPHI = b_{\phi\phi}$$

This subroutine must be supplied by the user to describe the particular body geometry to be considered in the calculation. The only programming requirements are that the COMMON block CBODY (also CBENT if bent nose is used) must be included in the subroutine and the above quantities must be defined for ϕ = PHI(M) and z = Z. The version of this routine supplied in the listings (Appendix D) is discussed in the User's Manual.

The entries BODYW and BODYR are used only once in the program (both are called from Section 1 of MAIN). The entry BODYR is used to read-in and compute the parameters used in the body shape function. The entry BODYW is used to print-out on the heading page a message which identifies the particular geometry being considered in the run.

12.2 FIELD

Calling sequence: CALL FIELD

Description: The purpose of this routine is to print-out the flow field data for a fixed axial station, Z, where Z is input from COMMON. The execution of this routine is controlled by the user with various output options (see the User's Manual for descriptions and instructions). This routine can be easily adapted to suit the individual needs of the user. Note that for each entry to the routine, the final (or corrected) values of the flow variables at the axial location Z are contained in the COMMON array P,D,U,V,W,ASQ. The specific outputs contained in the version of FIELD given in the listings are discussed in the Users' Manual.

12.3 OUT

Calling sequence: CALL OUT

Description: This routine is executed when the flow field calculation is completed. The purpose of the routine is to print out on-line surface pressure distributions and force and moment data. A description of these outputs is given in the User's Manual. The data to be printed out in this routine is read from the output tape (TAPE16) generated during the run. The routine converts the force and moment data to coefficient form and computes the centers of pressure (when defined) (see, Section 5 for definitions). The values of A_{ref}, z_c, z_{ref} used in the definitions of these quantities are input in this routine (see User's Manual for details).

12.4 RECOVR, SAVE

Calling sequence: EXTERNAL SAVE

CALL RECOVR (SAVE, FLAGS, CHECKSUM)

where

SAVE	is	the name of subroutine to be executed if flagged conditions occur
FLAGS	is	the octal value for conditions under which recovery code is to be executed. In this code, subroutine SAVE is executed if there is an arithmetic mode error, PP call or auto-recall error, or time or storage limit exceeded.
CHECKSUM		has to do with taking check sums. If equal to 0 no checksum desired.

Description: RECOVR is a special recovery routine supported by CDC on their operating systems SCOPE 3.4 and KRONOS 2.1. The RECOVR subroutine allows a user program to gain control at the time that abnormal job termination procedure would otherwise occur. In this program, subroutine SAVE is called.

Calling sequence: CALL SAVE (EX,ENRUN,RAPO)

where

EX	is	a 17 word integer array of the exchange package. The program does not use this array.
ENRUN	is	a flag that determines the type of program termination. The program does not use this flag.
RAPO		may be an array starting at RA + 1. The program does not use this array.

Description: The subroutine SAVE calls subroutine FIELD (see, Sec. 12.2) for the last IERRPR steps of the calculations. See the User's Manual for a description of the variable IERRPR. Also, subroutine SAVE calls subroutine OUT (see, Sec. 12.3).

12.5 TRANFD

Calling sequence: CALL TRANFD

Description: This routine is called when the mesh spacing in the radial direction is read in from cards (see user's manual for details). The routine is called only once in the program (from Section 1 of MAIN). TRANFD reads in the values of $\bar{x} = f(X_n)^*$ to be used in the calculation and computes numerically the derivatives f_X and f_{XX} (c.f., sec. 4.3). The quantities $\bar{x} = f$, f_X, and f_{XX} are returned using the COMMON arrays SFD, SFXD, and SFXXD, respectively. Note that NSFD is the number of radial points in the computational mesh.

12.6 TRANGD

Calling sequence: CALL TRANGD

Description: This routine is called when the mesh spacing in the ϕ-direction is read in from cards (see user's manual for details). The routine is called only once in the program (from Section 1 of MAIN). TRANGD reads in the values of $\phi = \phi_0\ g(Y_m)^{**}$ to be used in the calculation and computes numerically the derivatives g_Y and g_{YY} (see, sec. 4.3). The quantities g, g_Y, and g_{YY} are returned using the COMMON arrays SGD, SGYD, SGYYD respectively. Note that NSGD is the number of ϕ planes in the computational mesh. When a symmetric problem is being computed TRANGD also computes g_Y and g_{YY} on the fringe planes $-\Delta Y$ and $1+\Delta Y$. These quantities are returned using the COMMON variables GYMDY and GYYMDY (for $Y = -\Delta Y$) and GY1PDY and GYY1PDY (for $Y = 1+\Delta Y$).

*In this option, the mesh clustering function f(X,Y,Z) is assumed independent of Y and Z.

**In this option, the mesh clustering function g(Y,Z) is assumed independent of Z.

APPENDIX D

LISTINGS

In this appendix the fortran listings of the code are given (with the exception of RECOVR which is a CDC system routine). The listings given here contain the error mode update, IDENT DUMP, which is described in sec. 12.2 of the User's Manual.

```
      PROGRAM D3CSS(INPUT=1008,OUTPUT,TAPE3=512,TAPE16,TAPE17=512          D3CSS      2
     1   ,TAPE15,TAPE18=512,TAPE5=INPUT,TAPE6=OUTPUT,TAPE9)                D3CSS      3
C                                                                          D3CSS      4
C     THIS PROGRAM COMPUTES 3-D SUPERSONIC FLOW OVER BODIES                D3CSS      5
C     GIVEN BY R=B(PHI,Z)                                                  D3CSS      6
C     PRINCIPLE FEATURES ARE AS FOLLOWS                                    D3CSS      7
C       1.   WEAK CONSERVATION FORM FOR PDE'S ARE SOLVED, I.E.             D3CSS      8
C                 CU SUB Z  =  -CF SUB X  -  CG SUB Y  -  CE               D3CSS      9
C       2.   CONSERVATION DEPENDENT VARIABLES, CU, ARE RELATED TO THE       D3CSS     10
C            NON-CONSERVATION VARIABLES (PRESSURE (P), DENSITY (D),         D3CSS     11
C            VELOCITY COMPS (U,V,W)) BY THE FOLLOWING EQNS.                 D3CSS     12
C                 CU(1,N,M)=D*W(N,M)                                       D3CSS     13
C                 CU(2,N,M)=P(N,M)+D(N,M)*W(N,M)**2                        D3CSS     14
C                 CU(3,N,M)=D(N,M)*W(N,M)*U(N,M)                           D3CSS     15
C                 CU(4,N,M)=D(N,M)*W(N,M)*V(N,M)                           D3CSS     16
C       3.   MACCORMACK 2ND ORDER SCHEME USED AT INTERIOR PTS.             D3CSS     17
C       3.1  IF INTERIOR PRESSURE IS NON-POSITIVE, THEN THE                D3CSS     18
C            CONSERVATION QUANTITIES ARE REDEFINED USING AVERAGES           D3CSS     19
C            IN X DIRECTION                                                 D3CSS     20
C       4.   AT WALL, CHARACTERISTIC COMPATABILITY RELATIONS ARE USED       D3CSS     21
C            IN PREDICTOR-CORRECTOR MANNER                                  D3CSS     22
C                 NOTE THAT AT THE WALL                                     D3CSS     23
C            CU(1,1) STORES WALL LOG(P)                                     D3CSS     24
C            CU(2,1) STORES WALL ENTROPY                                    D3CSS     25
C            CU(3,1) STORES V2=V+(BPHI/B)*U                                 D3CSS     26
C       4.1  WALL ENTROPY RELAXATION IS AN OPTION                          D3CSS     27
C       5.   SHOCK SHAPE (C) AND SHOCK SLOPES (CZ) AND (CPHI) ARE          D3CSS     28
C            DETERMINE IN PREDICTOR-CORRECTOR MANNER.                       D3CSS     29
C                 NOTE THAT AT THE SHOCK (N=NC)                             D3CSS     30
C            CU(1,NC) STORES C                                              D3CSS     31
C            CU(2,NC) STORES CZ                                             D3CSS     32
C            CU(3,NC) STORES CPHI                                           D3CSS     33
C       6.   EITHER PERFECT GAS (CONSTANT GAMMA) OR REAL GAS               D3CSS     34
C            EQUILIBRIUM THERMO. CAN BE USED                                D3CSS     35
C       7.   MESH CLUSTERING TRANSFORMATIONS IN THE RADIAL AND             D3CSS     36
C            CIRCUMFERENTIAL DIRECTIONS ARE INCORPORATED                    D3CSS     37
C       8.   THE USER CAN SELECT EITHER OF TWO PROBLEMS -                  D3CSS     38
C                 (1)  THE SYMMETRIC PROBLEM                               D3CSS     39
C                 (2)  THE NON-SYMMETRIC PROBLEM                           D3CSS     40
C       8.1  IN THE SYMMETRIC PROBLEM - BODY IS SYMMETRIC WITH             D3CSS     41
C            RESPECT TO PITCH PLANE (ZERO YAW).   PHI=0 AND                 D3CSS     42
C            PHI=180 (DEGS) ARE SYMMETRY PLANES.  P,D,U,W ARE SYMM.         D3CSS     43
C            AND V IS ANTI-SYMM. AT SYMM. PLANES.                           D3CSS     44
C       8.2 IN THE NON-SYMMETRIC PROBLEM - BODY HAS NO                     D3CSS     45
C            RESTRICTIONS AND YAW MAY BE NON-ZERO.  ALL FLOW                D3CSS     46
C            VARIABLES ARE PERIODIC WITH PERIOD = 360 (DEGS)                D3CSS     47
C       9.   THIS VERSION HAS PROVISIONS FOR LOCAL EXPANSION AND            D3CSS     48
C            COMPRESSION JUMPS AT DISCONTINUITIES OF BZ AND/OR BPHI         D3CSS     49
C            IN Z DIRECTION                                                 D3CSS     50
C       10.  THIS VERSION HAS PROVISIONS FOR BENT NOSE BODY                 D3CSS     51
C            GEOMETRIES REQUIRING A SHIFT OF AXES (SEE                      D3CSS     52
C            USERS MANUAL FOR DETAILS)                                      D3CSS     53
C                                                                          D3CSS     54
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                           NEWCOM     1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                  NEWCOM     2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)       NEWCOM     3
      COMMON CU(4,20,25),CUP(4,20,25)                                      NEWCOM     4
```

```
C    ***    END OF BLANK COMMON    ***                                    CD3CSS     32
      COMMON /CBENT/ ZBB(25),ALNS,DST,AUON,BETA,RSN,CENUF,DELII,DELTA     NEWCOM      5
     1    ,COSBN,EPSQ,ZMAXS,PID2,TANBN,IBN,HN,THETABN                     CBENT       3
      COMMON /CTRANG/ NSGD,SGD(25),SGYD(25),SGYYD(25)                     NEWCOM      6
     1    ,GYMDY,GYYMDY,GY1PDY,GYY1PDY                                    CTRANG      3
     2    ,MCP                                                            NEWCOM      7
      COMMON /CTRANF/ NSFD,SFD(20),SFXD(20),SFXXD(20)                     NEWCOM      8
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                        CBODY       2
     1    ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)            CBODY       3
      COMMON /CBODYP/ DZ,PHI1J,PHI2J,BZT(3),BPHIT(3),BZZT(3)              CBODYP      2
     1    ,BPHPHT(3),BZPHIT(3),BZO(25),BPHIO(25),BPHPHO(25),BZPHIO(25)    CBODYP      3
     2    ,BZZO(25)                                                       CBODYP      4
      COMMON /CEVAL/ NCFL,JCFL,MCFL,CFL,BJ                                CEVAL       2
      COMMON /CDECODE/ GFF,GG,GIM1,HINF,V1INF,V2INF,WINX,DINF2            CDECODE     2
     1    ,ICHECK,X1K1NEG,X1K1NP2,GC(20,25)                               CDECODE     3
      COMMON /CINTEG/ FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ          CINTEG      2
     1    ,DYD3,MA,GY(25)                                                 CINTEG      3
      REAL MX,MY,MZ,MXZ,MYZ,MZZ                                           CINTEG      4
      COMMON /COUT/ ACH,ATTA,YAW,ZEND,XINDEF,VINF,SINF                    COUT        2
     1    ,NTARGET,TARGETZ(100)                                           COUT        3
      COMMON /CWALL/ PZ,V2Z,SZ,ISWMOD,MOD1,NJMPKT,NJMKTC,KCFL,KFAC        CWALL       2
     1    ,PZCOR(25)                                                      CWALL       3
      COMMON /CSAVE/ IERRPR,MAS                                           CSAVE       2
      COMMON /CSHOCK/ DCZ,DCZZ,DCPHZ,PDIF,SPDIF,DINX,D1INF,D2INF          CSHOCK      2
     1    ,UZCOR(4,25)                                                    CSHOCK      3
      COMMON /BLK01/ CZY(3),CPHIY(3),V2(3),VOWY(3),PWY(3),SWY(3)          BLK01       2
     1    ,UNOR(3,2),CF(4,20,2),CG(4,20,3),CE(4,20,2)                     BLK01       3
      COMMON /BLK02/ THETA,DY,TG4(3),TG5(3),TG6(3)                        BLK02       2
     1    ,X(20),XZ(20,2),XR(20,2),XPHI(20,2),Y(25)                       BLK02       3
     2    ,TF4(20,2),TF6(20,2),TF7(20,2)                                  BLK02       4
      COMMON /BLK03/ ICFL,A2(3),A3(3),A4(3),A5(3),A7(3)                   BLK03       2
     1    ,IJUMP(25),IJUMP1(25),IJMPKT(25)                                BLK03       3
      COMMON /BLK04/ GAMMA,GB,GD,GE,GA2,DDX,DDY,HOT2,ELIM,LCNT,ISWSMO,NA  BLK04       2
     1    ,SW(25),GM(20,25)                                               BLK04       3
      COMMON /RGASS/ AX,HX,TX,RRX,GX,NTEST,NGAS,NFIRST                    D3CSS      59
C                                                                         D3CSS      60
      DIMENSION KOUT(5),ZPRINT(5)                                         D3CSS      61
      EXTERNAL SAVE                                                       D3CSS      62
C                                                                         D3CSS      63
C      NCMAX MUST BE THE SAME NUMBER AS IN THE COMMON STATEMENTS          D3CSS      64
C          CORRESPONDING TO THE NUMBER OF RADIAL POINTS.                  D3CSS      65
C      MCMAX MUST BE THE SAME NUMBER AS IN THE COMMON STATEMENTS          D3CSS      66
C          CORRESPONDING TO THE NUMBER OF TANGENTIAL PLANES               D3CSS      67
C                                                                         D3CSS      68
      DATA (NCMAX=20),(MCMAX=25)                                          NEWCOM     22
C                                                                         D3CSS      70
      NAMELIST /INPUT1/ KA,ZEND,FACTOR,DZPRINT,KOUT,ZPRINT,IZONE,         D3CSS      71
     1   NCNEW,MCNEW,IPC,X1K1NEG,ISWSMO,ISWMOD,MOD1,                      D3CSS      72
     2   ZMOD1ON,ZMOD1OF,PHI1JD,PHI2JD,ZCFL1,ZCFL2,KCFL,KFAC,             D3CSS      73
     3   NJMPKT,NJMKTS,NTARGET,TARGETZ,IERRPR                             D3CSS      74
     4   ,ISTART,KSTART,ELIM,LCNT,IPRCFL,ISWDIF,NSGD,NSFD                 D3CSS      75
C                                                                         D3CSS      76
C*************************************************************************D3CSS      77
C                          SECTION 1                                      D3CSS      78
C              THIS SECTION IS EXECUTED ONLY ONCE,                        D3CSS      79
C                    AT INITIAL PROGRAM ENTRY.                            D3CSS      80
C*************************************************************************D3CSS      81
```

```
C                                                                       D3CSS      82
C  ***   READ IN INITIAL DATA AND PROGRAM CONTROLS   ***                 D3CSS      83
C     KA IS THE MAXIMUM NUMBER OF STEPS TO BE TAKEN                     D3CSS      84
C     ZEND IS THE LAST Z VALUE                                          D3CSS      85
C     FACTOR IS THE CFL FACTOR                                          D3CSS      86
C     DZPRINT IS THE Z INCREMENT USED FOR PRINTING                      D3CSS      87
C     KOUT(I) IS THE NUMBER OF STEPS BETWEEN PRINT OUTS WHEN            D3CSS      88
C            ZPRINT(I-1) .LE. Z .LT. ZPRINT(I)                          D3CSS      89
C     IZONE = 0 MEANS DO NOT REZONE                                     D3CSS      90
C           = 1 MEANS REZONE                                            D3CSS      91
C     NCNEW IS THE NUMBER OF POINTS IN RADIAL DIRECTION                 D3CSS      92
C     MCNEW IS THE NUMBER OF PLANES IN TANGENTIAL DIRECTION             D3CSS      93
C     IPC = 1   THEN FORWARD  DIFFERENCE FOR PREDICTOR STEP             D3CSS      94
C                    BACKWARD DIFFERENCE FOR CORRECTOR STEP             D3CSS      95
C     IPC = 0   THEN BACKWARD DIFFERENCE FOR PREDICTOR STEP             D3CSS      96
C                    FORWARD  DIFFERENCE FOR CORRECTOR STEP             D3CSS      97
C     K1K1NEG IS THE K IN THE 1-K-1 SMOOTHING OF                        D3CSS      98
C         CONSERVATION VECTORS IF P IS NEGATIVE                         D3CSS      99
C     ISWSMO = ISWSMO MEANS EXTRAPOLATE WALL ENTROPY FOR ISWSMO PLANES  D3CSS     100
C            = 0 MEANS NO EXTRAPOLATION OF WALL ENTROPY                 D3CSS     101
C     ISWMOD = 0 MEANS MOD 0 FOR WALL B.C.                              D3CSS     102
C            = 3 MEANS MOD 3 FOR WALL B.C.                              D3CSS     103
C     MOD1 = 1 MEANS SECOND ORDER ACCURACY AT WALL EXCEPT AFTER         D3CSS     104
C              DISCONTINUITIES IN BZ AND/OR BPHI                        D3CSS     105
C          = 0 MEANS NO SECOND ORDER ACCURACY                           D3CSS     106
C     ZMOD1ON IS THE Z VALUE AT WHICH TO TURN ON SECOND ORDER ACCURACY  D3CSS     107
C     ZMOD1OF IS THE Z VALUE AT WHICH TO TURN OFF SECOND ORDER ACCURACY D3CSS     108
C     (PHI1JD,PHI2JD) IS THE PHI OPEN INTERVAL IN WHICH NOT TO          D3CSS     109
C         USE SUBROUTINE JUMP                                           D3CSS     110
C     (ZCFL1,ZCFL2) IS THE Z OPEN INTERVAL TO USE FACTOR/KFAC           D3CSS     111
C         AS THE CFL FACTOR                                             D3CSS     112
C     KCFL IS THE NUMBER OF STEPS AFTER AN EXPANSION JUMP TO USE        D3CSS     113
C         FACTOR/KFAC AS THE CFL FACTOR                                 D3CSS     114
C     NJMPKT IS THE MAXIMUM NO OF STEPS AFTER AN EXPANSION JUMP         D3CSS     115
C         TO SET X DERIVATIVES TO ZERO AT WALL                          D3CSS     116
C     NJMKTS IS THE MAXIMUM NO OF STEPS AFTER A COMPRESSION JUMP        D3CSS     117
C         TO SET X DERIVATIVES TO ZERO AT WALL                          D3CSS     118
C     NTARGET IS THE NUMBER OF Z TARGET POINTS                          D3CSS     119
C     TARGETZ IS THE ARRAY OF THE Z TARGET POINTS                       D3CSS     120
C     IERRPR - THE LAST IERRPR STEPS WILL BE PRINTED IF AN ERROR OCCURS D3CSS     121
C     ISTART = 1 MEANS RESTART FROM TAPE15                              D3CSS     122
C            = 0 MEANS DO NOT RESTART FROM TAPE15                       D3CSS     123
C     KSTART IS THE K STEP NO. ON TAPE15 AT WHICH TO RESTART            D3CSS     124
C     ELIM AND LCNT ARE ERROR AND NUMBER LIMITS ON ITERATIVE PROCEDURES D3CSS     125
C     IPRCFL IS THE NO. OF STEPS BETWEEN PRINTOUTS OF CFL INFORMATION   D3CSS     126
C     ISWDIF = 1 THEN THE DIFFERENCING IS SWITCHED FROM STEP TO STEP    D3CSS     127
C            = 0 THEN DIFFERENCING IS NOT SWITCHED                      D3CSS     128
C     NSGD IS THE NO. OF PHI"S TO BE READ                               D3CSS     129
C     SGD IS THE ARRAY OF PHI"S (DEGREES) READ IN SUBROUTINE TRANGD     D3CSS     130
C     NSFD IS THE NO. OF SF(X)"S TO BE READ IN SUBROUTINE TRANFD        D3CSS     131
C     SFD IS THE ARRAY OF SF(X)"S READ IN SUBROUTINE TRANFD             D3CSS     132
C                                                                       D3CSS     133
      PI=4.*ATAN(1.) $ RAD=PI/180.                                      D3CSS     134
      DO 5 I=1,5                                                        D3CSS     135
      KOUT(I)=20 $ ZPRINT(I)=1000000.                                   D3CSS     136
    5 CONTINUE                                                          D3CSS     137
      KA=2000 $ FACTOR=.9 $ DZPRINT=1000000.                            D3CSS     138
```

```
      IZONE=0 $ IPC=0                                                    D3CSS     139
      K1K1NEG=2 $ ISWSMO=0 $ ISWMOD=3 $ MOD1=1                           D3CSS     140
      ZMOD1ON=1000000. $ ZMOD1OF=1000000. $ PHI1JD=0. $ PHI2JD=0.        D3CSS     141
      ZCFL1=0. $ ZCFL2=0. $ KCFL=0 $ KFAC=3 $ NJMPKT=0                   D3CSS     142
      NJMKTS=4 $ NTARGET=0 $ ISTART=0 $ KSTART=0 $ IERRPR=-1             D3CSS     143
      IPRCFL=1 $ ISWDIF=0 $ NSGD=0 $ NSFD=0                              D3CSS     144
      ELIM=.001 $ LCNT=20 $ IBN=0 $ HN=0.                                D3CSS     145
      READ (5,INPUT1)                                                   D3CSS     146
      CALL BODYR                                                        D3CSS     147
      PHI(1)=0.                                                         D3CSS     148
      CALL OUTR                                                         D3CSS     149
      ICP=1-IPC $ MOD1ON=0                                              D3CSS     150
      X1K1NEG=K1K1NEG $ X1K1NP2=X1K1NEG+2.                              D3CSS     151
      PHI1J=PHI1JD*RAD $ PHI2J=PHI2JD*RAD                               D3CSS     152
      ICFL=0 $ FACJMP=FACTOR/KFAC $ FACT1=FACJMP $ KFAC1=1              D3CSS     153
      NJMKTC=KFAC*NJMKTS $ ITARGET=1                                    D3CSS     154
C                                                                       D3CSS     155
C       NC IS THE NUMBER OF POINTS IN RADIAL DIRECTION                  D3CSS     156
C       MC IS THE NUMBER OF PLANES IN TANGENTIAL DIRECTION              D3CSS     157
C       NA IS THE NUMBER OF INTERVALS IN RADIAL DIRECTION               D3CSS     158
C       MA IS THE NUMBER OF INTERVALS IN TANGENTIAL DIRECTION           D3CSS     159
C       ATTACK IS THE ANGLE OF ATTACK IN DEGREES                        D3CSS     160
C       YAW IS THE SIDESLIP ANGLE IN DEGREES                            D3CSS     161
C       ACH IS THE MACH NUMBER                                          D3CSS     162
C       K IS THE NUMBER OF STEPS IN THE AXIAL DIRECTION                 D3CSS     163
C                                                                       D3CSS     164
      IF (IERRPR .GE. 0) CALL RECOVR(SAVE,78,0)                         D3CSS     165
      IERRPR=IABS(IERRPR)                                               DUMP        1
      IF (ISTART .EQ. 0) GO TO 9                                        D3CSS     166
    7 READ (15) NC,MC,ATTACK,YAW,ACH,GAMMA,PINF,DINF,PHIO,K,Z           D3CSS     167
     A    ,NGAS,NTEST,RRX                                             D3CSS     168
     1    ,FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ                  D3CSS     169
     2    ,(PHI(M),C(M),CZ(M),CPHI(M),M=1,MC)                         D3CSS     170
     3    ,((R(N,M),U(N,M),V(N,M),W(N,M),P(N,M),D(N,M),M=1,MC),N=1,NC) D3CSS     171
      IF (EOF(15)) 12,8                                                 D3CSS     172
    8 WRITE (16) NC,MC,ATTACK,YAW,ACH,GAMMA,PINF,DINF,PHIO,K,Z          D3CSS     173
     A    ,NGAS,NTEST,RRX                                             D3CSS     174
     1    ,FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ                  D3CSS     175
     2    ,(PHI(M),C(M),CZ(M),CPHI(M),M=1,MC)                         D3CSS     176
     3    ,((R(N,M),U(N,M),V(N,M),W(N,M),P(N,M),D(N,M),M=1,MC),N=1,NC) D3CSS     177
      IF (K .LT. KSTART) GO TO 7                                        D3CSS     178
      GO TO 15                                                          D3CSS     179
    9 READ (3) NC,MC,ATTACK,YAW,ACH,GAMMA,PINF,DINF,PHIO,K,Z            D3CSS     180
     A   ,NGAS,NTEST,RRX                                              D3CSS     181
     1   ,FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ                   D3CSS     182
     2    ,(PHI(M),C(M),CZ(M),CPHI(M),M=1,MC)                         D3CSS     183
     3    ,((R(N,M),U(N,M),V(N,M),W(N,M),P(N,M),D(N,M),M=1,MC),N=1,NC) D3CSS     184
      IF (EOF(3)) 10,15                                                 D3CSS     185
   10 WRITE (6,4000)                                                    D3CSS     186
 4000 FORMAT(*1  NO DATA ON TAPE3  ---  STOP  ---*)                     D3CSS     187
      STOP                                                              D3CSS     188
   12 WRITE (6,4005) K,KSTART $ STOP                                    D3CSS     189
 4005 FORMAT(1H1,*THE LAST K ON TAPE15 IS*,I5,* LESS THAN*,I5,          D3CSS     190
     1    *  ---  STOP  ---*)                                          D3CSS     191
C                                                                       D3CSS     192
C   ***   PARAMETERS   ***                                            D3CSS     193
   15 ATTA=ATTACK $ ATTACK=ATTACK*RAD $ DZ=0.                           D3CSS     194
```

```
      IF (NSGD .GT. 0) CALL TRANGD                                          D3CSS     195
      IF (NSFD .GT. 0) CALL TRANFD                                          D3CSS     196
      SINAL=SIN(ATTACK) $ COSAL=COS(ATTACK)                                 D3CSS     197
      YA=YAW*PI/180. $ SINBET=SIN(YA) $ COSBET=COS(YA)                      D3CSS     198
      PDIF=PINF/DINF                                                        D3CSS     199
      DINF2=DINF*DINF                                                       D3CSS     200
      NFIRST=100                                                            D3CSS     201
      GX=GAMMA                                                              D3CSS     202
      SINF=0.                                                               D3CSS     203
      IF(NTEST.GE.0)GO TO 50                                                D3CSS     204
      CALL RGAS(PINF,DINF,SINF,4)                                           D3CSS     205
      HINF=HX                                                               D3CSS     206
      GAMMA=1./(1.-PINF/(DINF*HINF))                                        D3CSS     207
      VINF=AX*ACH                                                           D3CSS     208
      GO TO 60                                                              D3CSS     209
   50 VINF=SQRT(GAMMA*PINF/DINF)*ACH                                        D3CSS     210
      HINF=GAMMA*PINF/((GAMMA-1.)*DINF)                                     D3CSS     211
   60 HOT2=2.*HINF+VINF**2                                                  D3CSS     212
      GIM1=1.-GAMMA                                                         D3CSS     213
      GI1=GAMMA                                                             D3CSS     214
      GB=1./(GAMMA-1.)                                                      D3CSS     215
      GA=GB*GAMMA $ GA2=2.*GA                                               D3CSS     216
      GD=.5/GB                                                              D3CSS     217
      GE=GD+1.                                                              D3CSS     218
      GFF=GAMMA+1.                                                          D3CSS     219
      GG=GFF/GB                                                             D3CSS     220
      WINX=COSBET*COSAL*VINF                                                D3CSS     221
      V1INF=COSBET*SINAL*VINF $ V2INF=VINF*SINBET                           D3CSS     222
      SPDIF=1./SQRT(PDIF)                                                   D3CSS     223
      DINX=WINX*SPDIF $ D1INF=V1INF*SPDIF $ D2INF=V2INF*SPDIF               D3CSS     224
      IF (NSFD .LE. 0) GO TO 17 $ IF (NSFD .EQ. NC) GO TO 17                D3CSS     225
      NCNEW=NSFD $ IF (IZONE .NE. 0) GO TO 17 $ IZONE=1 $ MCNEW=MC          D3CSS     226
   17 CONTINUE                                                              D3CSS     227
      IF (NSGD .LE. 0) GO TO 16 $ IF (NSGD .EQ. MC) GO TO 16                D3CSS     228
      MCNEW=NSGD $ IF (IZONE .NE. 0) GO TO 16 $ IZONE=1 $ NCNEW=NC          D3CSS     229
   16 IF (IZONE .NE. 0)                                                     D3CSS     230
     1CALL REZONE(NCNEW,MCNEW,ASQ,IJUMP,0,BZ0,BPHI0,RZZ0,XZ,NCMAX,MCMAX)   D3CSS     231
      NA=NC-1 $ MCP=MC+1 $ MA=MC-1 $ MCP2=MC+2                              D3CSS     232
      MCM2=MA-1 $ IIC=4                                                     D3CSS     233
      DX=1./NA $ DY=1./MA $ DDY=MA $ DDX=NA $ BJ=DX/DY                      D3CSS     234
      IF (IPC .EQ. 0) BJ=-BJ                                                D3CSS     235
      DY03=DY/3.                                                            D3CSS     236
      IF (PHI0 .LE. 2.*PI-1.E-6) GO TO 18                                   D3CSS     237
C   ***   SET VARIABLES FOR NON-SYMMETRIC PROBLEM (PHI0=360)   ***          D3CSS     238
      COSPHI(MC)=1. $ SINPHI(MC)=0.                                         D3CSS     239
      MAS=MA $ MCS=MC $ IDYAW=1 $ MCM2S=MCM2                                D3CSS     240
      GO TO 21                                                              D3CSS     241
C   ***   SET VARIABLES FOR SYMMETRIC PROBLEM (PHI0=180)   ***             D3CSS     242
   18 MAS=MC $ MCS=MCP $ IDYAW=0 $ MCM2S=0                                  D3CSS     243
   21 Y0=-DY $ YMCP=1.+DY                                                   D3CSS     244
C                                                                           D3CSS     245
C   ***   PRINT THE HEADING PAGE   ***                                      D3CSS     246
      PHI0D=PHI0/RAD                                                        D3CSS     247
      WHEN=DATE(WHEN) $ CLTIM=TIME(CLTIM)                                   D3CSS     248
      IVERSON=10                                                            D3CSS     249
      DO 150 I=1,3                                                          D3CSS     250
      WRITE (6,3000) IVERSON,WHEN,CLTIM                                     D3CSS     251
```

```
      WRITE (6,3010)                                                    D3CSS     252
      WRITE (6,3020) KA,ZEND,FACTOR                                     D3CSS     253
      WRITE(6,3021)ELIM,LCNT                                            D3CSS     254
      WRITE (6,3030) ZPRINT                                             D3CSS     255
      WRITE (6,3040) KOUT                                               D3CSS     256
      WRITE (6,3045) DZPRINT                                            D3CSS     257
      WRITE (6,3050) ACH,ATTA,YAW,VINF                                  D3CSS     258
      WRITE (6,3051) PINF,DINF,HINF,HOT2/2.,SINF                        D3CSS     259
      IF (NGAS .LE. 0) WRITE (6,3054) GAMMA,RRX                         CORR1       1
      IF (NGAS .GT. 0) WRITE (6,3056) NGAS                              D3CSS     261
      WRITE (6,3060) PHIOD                                              D3CSS     262
      WRITE (6,3070) Z                                                  D3CSS     263
      WRITE (6,3080) NA,MA                                              D3CSS     264
      IF (IZONE .NE. 0) WRITE (6,3090)                                  D3CSS     265
      CALL BODYW(IDUM)                                                  D3CSS     266
      CALL TRANGW(DUM,IDUM,IDUM)                                        D3CSS     267
      CALL TRANFW(IDUM,IDUM,IDUM)                                       D3CSS     268
      WRITE (6,4010)                                                    D3CSS     269
 4010 FORMAT(//////,25X,*ADDITIONAL FEATURES*)                          D3CSS     270
      IF (IPC .EQ. 1) WRITE (6,3108)                                    D3CSS     271
      IF (IPC .EQ. 0) WRITE (6,3109)                                    D3CSS     272
      IF (ISWSMO .NE. 0) WRITE (6,4019) ISWSMO                          D3CSS     273
      IF (ISWSMO .EQ. 0) WRITE (6,4020)                                 D3CSS     274
      IF (ISWMOD .EQ. 0) WRITE (6,4029)                                 D3CSS     275
      IF (ISWMOD .EQ. 3) WRITE (6,4030)                                 D3CSS     276
      IF (ZMOD1ON .LE. ZEND) WRITE (6,4035) ZMOD1ON                     D3CSS     277
      IF (MOD1 .EQ. 1) WRITE (6,4040) ZMOD1OF                           D3CSS     278
      WRITE (6,3206) K1K1NEG                                            D3CSS     279
      WRITE (6,3152) PHI1JD,PHI2JD                                      D3CSS     280
      WRITE (6,3135) FACJMP,ZCFL1,ZCFL2                                 D3CSS     281
      WRITE (6,3136) FACJMP,KCFL                                        D3CSS     282
      WRITE (6,3147) NJMPKT,NJMKTS                                      D3CSS     283
      IF (ISWDIF .NE. 0) WRITE (6,4055)                                 D3CSS     284
  150 CONTINUE                                                          D3CSS     285
      IF (Z .GT. ZCFL2) ZCFL1=2.*ZEND                                   D3CSS     286
 3000 FORMAT(1H1,20X,*PROGRAM D3CSS*,6X,*VERSION*,I4,6X,*DATE*,A12,6X,  D3CSS     287
     1            *TIME*,A12)                                          D3CSS     288
 3010 FORMAT(11X,*3-D SUPERSONIC FLOW  -  *,                            D3CSS     289
     1   *FLOW IS NONSYMMETRICAL*)                                      D3CSS     290
 3020 FORMAT(11X,*MAXIMUM NO. OF STEPS =*,I5,6X,                        D3CSS     291
     1          *LAST Z VALUE =*,1PE15.6,6X,*CFL FACTOR =*,0PF6.3)     D3CSS     292
 3021 FORMAT(11X,*ERROR LIMIT *,1PE12.4,2X,*MAXIMUM NUMBER OF ITERATIONS D3CSS     293
     1*,I5)                                                             D3CSS     294
 3030 FORMAT(11X,*PRINT CONTROLS ARE*,5X,*ZPRINT*,5F12.2)               D3CSS     295
 3040 FORMAT(11X,23X,*KOUT  *,5I12)                                     D3CSS     296
 3045 FORMAT(11X,23X,*DZPRINT*,F11.2)                                   D3CSS     297
 3050 FORMAT(11X,*MACH NO. =*,F8.2,6X,*ANGLE OF ATTACK =*,F7.2,6X,      D3CSS     298
     1   *YAW ANGLE =*,F7.2,3X,*VINF =*,F10.2)                          D3CSS     299
 3051 FORMAT(11X,*FREE STREAM PROPERTIES , PINF = *,1PE12.4,* DINF = *  D3CSS     300
     1   ,1PE12.4,* HINF =*,1PE13.4,* HO =*,1PE13.4,* SINF =*,1PE13.4)  D3CSS     301
 3054 FORMAT(11X,*PERFECT GAS   (GAMMA =*,F6.2,4X,                      CORR1       2
     1   *GAS CONSTANT =*,1PE15.6,*)*)                                  CORR1       3
 3056 FORMAT(11X,*REAL GAS   (GAS NUMBER IS*,I3,*)*)                    D3CSS     303
 3060 FORMAT(11X,*FLOW IS PERIODIC WITH PERIOD =*,F7.2)                 D3CSS     304
 3070 FORMAT(11X,*CALC. BEGINS AT Z =*,E15.7)                           D3CSS     305
 3080 FORMAT(11X,*RADIAL INTERVALS NA =*,I4,6X,                         D3CSS     306
     1           *TANGENTIAL INTERVALS MA =*,I4)                        D3CSS     307
```

```
 3090 FORMAT(11X,*REZONE THE MESH IN THIS RUN*)                          D3CSS     308
 3108 FORMAT(1H0,10X,*FORWARD DIFFERENCE FOR PREDICTOR STEP AND *,       D3CSS     309
     1   *BACKWARD DIFFERENCE FOR CORRECTOR STEP IN X DIRECTION*)        D3CSS     310
 3109 FORMAT(1H0,10X,*BACKWARD DIFFERENCE FOR PREDICTOR STEP AND *,      D3CSS     311
     1   *FORWARD DIFFERENCE FOR CORRECTOR STEP IN X DIRECTION*)         D3CSS     312
 3135 FORMAT(1H0,10X,*THE CFL FACTOR IS REDUCED TO*,0PF6.3,              D3CSS     313
     1   * WHEN Z IS IN THE INTERVAL (*,F8.2,*,*,F8.2,*)*)               D3CSS     314
 3136 FORMAT(1H0,10X,*USE CFL FACTOR =*,0PF6.3,                          D3CSS     315
     1   * FOR*,I3,* STEPS AFTER AN EXPANSION JUMP OCCURS*)              D3CSS     316
 3147 FORMAT(1H0,10X,*THE TERMS FOR X DERIVATIVES AT THE WALL *,         PXWALL      1
     1*ARE MODIFIED FOR*,/,                                              PXWALL      2
     2   17X,I3,* STEPS AFTER AN EXPANSION JUMP AND*,I3,                 D3CSS     319
     3   * STEPS AFTER A COMPRESSION JUMP*)                              D3CSS     320
 3152 FORMAT(1H0,10X,*USING JUMP WHICH COMPUTES JUMPS CORRESPONDING *,   D3CSS     321
     1   *TO DISCONTS. IN BZ AND/OR BPHI EXCEPT FOR THE PHI INTERVAL (*, D3CSS     322
     2   F7.2,*,*,F7.2,*)*)                                              D3CSS     323
 3206 FORMAT(1H0,10X,*IF PRESSURE IS NEGATIVE THEN THE CONSERVATION *,   D3CSS     324
     1   *VECTORS ARE SMOOTHED BY 1-*,I2,*-1*)                           D3CSS     325
 4019 FORMAT(1H0,10X,*WALL ENTROPY EXTRAPOLATION FOR*,I3,* PLANES*,      D3CSS     326
     1   * UNTIL A COMPRESSION JUMP AND THEN NO EXTRAPOLATION*)          CORR1       4
 4020 FORMAT(1H0,10X,*NO WALL ENTROPY EXTRAPOLATION*)                    D3CSS     328
 4029 FORMAT(1H0,10X,*MOD 0 FOR WALL POINTS*)                            D3CSS     329
 4030 FORMAT(1H0,10X,*MOD 3 FOR WALL POINTS UNTIL A JUMP OCCURS AND *,   D3CSS     330
     1   *THEN MOD 0 IS USED*)                                           D3CSS     331
 4035 FORMAT(1H0,10X,*OPTION FOR SECOND ORDER ACCURACY AT WALL POINTS *, D3CSS     332
     1   *IS TURN ON AT Z =*,1PE15.6)                                    D3CSS     333
 4040 FORMAT(1H0,10X,*SECOND ORDER ACCURACY IS USED AT WALL POINTS *,    D3CSS     334
     1   *FOR Z LESS THAN*,1PE15.6,* OR UNTIL JUMP IS CALLED*)           D3CSS     335
 4055 FORMAT(1H0,10X,*THE DIFFERENCING (FORWARD - BACKWARD) IS *         D3CSS     336
     1   ,*SWITCHED FROM STEP TO STEP*)                                  D3CSS     337
C                                                                        D3CSS     338
C   ***   INITIALIZATIONS   ***                                          D3CSS     339
      DO 25 N=1,NC                                                       D3CSS     340
   25 X(N)=(N-1)/DDX                                                     D3CSS     341
      DO 36 M=1,MAS                                                      D3CSS     342
      IJUMP1(M)=IJMPKT(M)=0                                              D3CSS     343
      Y(M)=(M-1)/DDY                                                     D3CSS     344
      DO 35 N=1,NC                                                       D3CSS     345
      TDNM=D(N,M) $ TPNM=P(N,M)                                          D3CSS     346
      CALL RGAS(TPNM,TDNM,DUMM,4)                                        D3CSS     347
      W(N,M)=TWNM=SQRT(HOT2-2.*HX-U(N,M)**2-V(N,M)**2)                   D3CSS     348
      TCU=CU(1,N,M)=TDNM*TWNM                                            D3CSS     349
          CU(2,N,M)=TPNM+TWNM*TCU                                        D3CSS     350
          CU(3,N,M)=U(N,M)*TCU                                           D3CSS     351
          CU(4,N,M)=V(N,M)*TCU                                           D3CSS     352
      GM(N,M)=1./(1.-TPNM/(TDNM*HX))                                     D3CSS     353
      GC(N,M)=1.E+99                                                     D3CSS     354
   35 ASQ(N,M)=AX*AX                                                     D3CSS     355
   36 CONTINUE                                                           D3CSS     356
      IF (IDYAW .EQ. 0) CPHI(1)=CPHI(MC)=0.                              D3CSS     357
C                                                                        D3CSS     358
C*********************************************************************** D3CSS     359
C          ***  PRELIMINARY PREDICTOR LOOP (INITIAL STEP ONLY)  ***     D3CSS     360
C        IN THIS LOOP, K1 CORRESPONDS TO M-1 AND K2 TO M IN CF,CG,CE     D3CSS     361
C            CONSERVATION VECTORS, AND TRANF, TRANG QUANTITIES.          D3CSS     362
C        NOTE WELL.  IN THIS LOOP CUP AND CP STORES PREDICTED            D3CSS     363
C             Z-DIFFERENCES NOT PREDICTED VALUES.                        D3CSS     364
```

```
      K1=1 $ K2=2                                                          D3CSS     365
      CFL=0.0                                                              D3CSS     366
      IF(IBN.EQ.0) GO TO 40                                                D3CSS     367
      TCR=BSN/COSBN $ IF (Z .GT. DST) HN=0.                                SHFAX2      1
      CCC=(1.-TCR)**2                                                      D3CSS     369
   40 CONTINUE                                                             D3CSS     370
      DO 100 M=1,MCS                                                       D3CSS     371
      KK=K1 $ K1=K2 $ K2=KK                                                D3CSS     372
      IF (M .LT. MCS) GO TO 69                                             D3CSS     373
      IF (IDYAW .EQ. 1) GO TO 65                                           D3CSS     374
C     ***   COMPUTE CG VECTOR AT Y=1+DY USING SYMMETRY CONDITIONS    ***   D3CSS     375
      CALL TRANG(YMCP,MCP,K2)                                              D3CSS     376
      CALL EVALSY(DUM,MA,K1,K2,K2,DUM)                                     D3CSS     377
      GO TO 80                                                             D3CSS     378
C     ***   COMPUTE CG VECTOR AT Y=1    ***                              D3CSS     379
   65 CALL TRANG(1.,MC,K2)                                                 D3CSS     380
      CALL EVALPR(DUM,1,DUM,K2,K2,DUM)                                     D3CSS     381
      GO TO 80                                                             D3CSS     382
   69 CALL TRANG(Y(M),M,K2)                                                D3CSS     383
      PHI(M)=THETA*PHIO $ GY(M)=TG4(K2)                                    D3CSS     384
      COSPHI(M)=COS(PHI(M)) $ SINPHI(M)=SIN(PHI(M))                        D3CSS     385
      IF (IBN .EQ. 1) GO TO 1045                                           D3CSS     386
      ZBB(M)=1.E08 $ GO TO 1047                                            D3CSS     387
C     ***   COMPUTE SPHERE CONE JUNCTURE FOR BENT NOSE    ***              D3CSS     388
 1045 SCR=(COSPHI(M)*TANBN)**2                                             D3CSS     389
      AAB=1.+SCR                                                           D3CSS     390
      BBC=2.*(TCR-1.-SCR)                                                  D3CSS     391
      IF(PHI(M).LT.PID2.OR.PHI(M).GT.PI+PID2) GO TO 1046                   D3CSS     392
      ZBB(M)=(-BBC+SQRT(BBC*BBC-4.*AAB*CCC))/(2.*AAB)                      D3CSS     393
      GO TO 1047                                                           D3CSS     394
 1046 ZBB(M)=(-BBC-SQRT(BBC*BBC-4.*AAB*CCC))/(2.*AAB)                      D3CSS     395
 1047 CONTINUE                                                             D3CSS     396
      CALL BODY(M)                                                         D3CSS     397
      BZO(M)=BZ(M) $ BPHIO(M)=BPHI(M)                                      D3CSS     398
      BZZO(M)=BZZ $ BZPHIO(M)=BZPHI $ BPHPHO(M)=BPHPHI                     D3CSS     399
      CALL BODYP(M,K2)                                                     D3CSS     400
C     ***   INITIALIZATION OF CU(I,1,M),I=1,2,3    ***                   D3CSS     401
      PM=P(1,M) $ CU(1,1,M)=ALOG(PM)                                       D3CSS     402
      CALL RGAS(PM,D(1,M),SW(M),4)                                         D3CSS     403
      ASQ(1,M)=AX*AX                                                       D3CSS     404
      CU(2,1,M)=SW(M)                                                      D3CSS     405
      CU(3,1,M)=V(1,M)+U(1,M)*BPHI(M)/R(M)                                 D3CSS     406
      CU(1,NC,M)=C(M)  $ CU(2,NC,M)=C7(M)  $ CU(3,NC,M)=CPHI(M)            D3CSS     407
      CU(4,NC,M)=CUP(4,NC,M)=0.                                            D3CSS     408
      CALL RGAS(P(NC,M),D(NC,M),DUMMY,4)                                   D3CSS     409
      ASQ(NC,M)=AX*AX                                                      D3CSS     410
      CALL TRANF(M,K2,K2)                                                  D3CSS     411
      CALL EVAL(0,M,K2,K2,K2,K2)                                           D3CSS     412
      IF (M .EQ. 1) GO TO 100                                              D3CSS     413
   80 MB=M-1                                                               D3CSS     414
      DO 90 N=2,NA $ NP=N+IPC                                              D3CSS     415
      DO 90 I=1,4                                                          D3CSS     416
      CUP(I,N,MB)=-(CF(I,NP,K1)-CF(I,NP-1,K1))*DDX                         D3CSS     417
     1           -(CG(I,N,K2)-CG(I,N,K1))*DDY-CE(I,N,K1)                   D3CSS     418
   90 CONTINUE                                                             D3CSS     419
      CALL WALL(MB,K2,K1,K1,K1,0)                                          D3CSS     420
      CUP(1,1,MB)=PZ $ CUP(3,1,MB)=V27                                     D3CSS     421
```

```
      CUP(2,1,MB)=SZ                                                    D3CSS    422
      CALL SHOCK(MB,K2,K1,K1,K1,0)                                      D3CSS    423
      CUP(1,NC,MB)=DCZ $ CUP(2,NC,MB)=DCZZ $ CUP(3,NC,MB)=DCPHZ         D3CSS    424
  100 CONTINUE                                                          D3CSS    425
      IF (ISWDIF .NE. 0) IPC=ICP                                        D3CSS    426
      IF (IDYAW .EQ. 1) GO TO 125                                       D3CSS    427
C   ***   SET SYMMETRY CONDITIONS   ***                                D3CSS    428
      DO 110 N=2,NC                                                     D3CSS    429
  110 CU(4,N,1)=CU(4,N,MC)=0.                                           D3CSS    430
      CU(3,1,1)=CU(3,1,MC)=CU(3,NC,1)=CU(3,NC,MC)=0.                    D3CSS    431
C*********************************************************************D3CSS    432
C                                                                       D3CSS    433
C   ***   PRINT THE INITIAL INPUT DATA   ***                            D3CSS    434
  125 IPRINT=1 $ CALL FIELD $ PRINTZ=Z $ CALL INTEG(0)                  D3CSS    435
      IF (ISTART .EQ. 0) GO TO 725                                      D3CSS    436
      GO TO 750                                                         D3CSS    437
C                                                                       D3CSS    438
C*********************************************************************D3CSS    439
C                          SECTION 2                                    D3CSS    440
C        THIS SECTION CONTAINS THE MAIN CALCULATION LOOP                D3CSS    441
C*********************************************************************D3CSS    442
C                                                                       D3CSS    443
  200 K=K+1                                                             D3CSS    444
C                                                                       D3CSS    445
C   ***   COMPUTE DZ AND UPDATE CUP(N,M) AND CP(M)   ***                D3CSS    446
      IF (Z .LE. ZCFL1) GO TO 202                                       D3CSS    447
      ZCFL1=ZCFL2 $ ZCFL2=2.*ZEND                                       D3CSS    448
      DUM=FACTOR $ FACTOR=FACT1 $ FACT1=DUM                             D3CSS    449
      IDUM=KFAC $ KFAC=KFAC1 $ KFAC1=IDUM                               D3CSS    450
  202 FAC=FACTOR                                                        D3CSS    451
      IF (ICFL .LE. 0) GO TO 203 $ IF (KCFL .LE. 0) GO TO 203           D3CSS    452
      ICFL=ICFL+1 $ IF (ICFL .GT. KCFL) ICFL=0                          D3CSS    453
      FAC=FACJMP                                                        D3CSS    454
  203 DZ=FAC*DX/CFL                                                     D3CSS    455
      IF (DZ .GE. 1.E-04) GO TO 205                                     D3CSS    456
      WRITE (6,3990) $ CALL SAVE(DUM,DUM,DUM)                           D3CSS    457
 3990 FORMAT(1H1,*DZ IS LESS THAN 1.E-4   ---   STOP   ---*)            D3CSS    458
  205 IPRINT=0 $ ZZ=Z+DZ                                                D3CSS    459
  208 IF (ITARGET .GT. NTARGET) GO TO 210                               D3CSS    460
      IF (ZZ .LT. TARGETZ(ITARGET)) GO TO 210                           D3CSS    461
      ITARGET=ITARGET+1                                                 D3CSS    462
      IF (TARGETZ(ITARGET-1)-Z .LE. 0.) GO TO 208                       D3CSS    463
      IPRINT=1                                                          D3CSS    464
  210 IF (MOD(K,IPRCFL) .EQ. 0) WRITE (6,3672) K,DZ,CFL,NCFL,MCFL,JCFL,Z D3CSS    465
 3672 FORMAT(1H0,*K IS*,I5,5X,*DZ IS*,1PE15.7,5X,*CFL IS*,1PE15.7,5X,   D3CSS    466
     1       *NCFL IS*,I3,5X,*MCFL IS*,I3,5X,*JCFL IS*,I3,5X,           D3CSS    467
     2       *Z IS*,1PE15.7)                                            D3CSS    468
      IF(HN.EQ.0) GO TO 209                                             D3CSS    469
C   ***   CHECK FOR AXIS SHIFT FOR BENT NOSE   ***                      D3CSS    470
      IF(DZ*TAN(THETABN-ALNS).LT.B(INT(FLOAT(MC)*PI/PHIO))-CENUF) GO TO D3CSS    471
     1 207                                                              D3CSS    472
      IA=1 $ GO TO 213                                                  D3CSS    473
  207 IF (Z .LT. DST) GO TO 209                                         D3CSS    474
      IA=2                                                              D3CSS    475
  213 CALL FIELD                                                        D3CSS    476
      IF (IDYAW .EQ. 0) GO TO 212                                       D3CSS    477
      PHI(MC)=PHIO $ C(MC)=C(1) $ CZ(MC)=CZ(1) $ CPHI(MC)=CPHI(1)       D3CSS    478
```

```
      DO 211 N=1,NC                                                    D3CSS     479
      R(N,MC)=R(N,1) $ U(N,MC)=U(N,1) $ V(N,MC)=V(N,1)                 D3CSS     480
      P(N,MC)=P(N,1) $ D(N,MC)=D(N,1)                                  D3CSS     481
  211 CONTINUE                                                         D3CSS     482
  212 CALL SHFAXD(IA,NC,MC,CU,CUP,Y,GY,PZCOR,R,RZ,BPHI)                D3CSS     483
      ATTACK=ATTA                                                      D3CSS     484
      GO TO 15                                                         D3CSS     485
  209 CONTINUE                                                         D3CSS     486
      Z=ZZ                                                             D3CSS     487
C   ***   CHECK TO TURN OFF SECOND ORDER ACCURACY   ***                D3CSS     488
      IF (Z .GE. ZMOD1OF .AND. ZMOD1OF .GT. ZMOD1ON) MOD1=0            D3CSS     489
      IF (Z .GE. ZMOD1OF .AND. MOD1ON .EQ. 0) MOD1=0                   D3CSS     490
C   ***   CHECK TO TURN ON SECOND ORDER ACCURACY   ***                 D3CSS     491
      IF (MOD1ON .EQ. -1) MOD1=1                                       D3CSS     492
      IF (MOD1ON .EQ. -1) MOD1ON=1                                     D3CSS     493
      IF (Z .LT. ZMOD1ON) GO TO 250 $ IF (MOD1ON .EQ. 1) GO TO 250     D3CSS     494
      MOD1ON=-1 $ IF (MOD1 .EQ. 1) MOD1ON=1                            D3CSS     495
  250 DZDX=DZ*DDX $ DZDY=DZ*DDY                                        D3CSS     496
C   ***   PREDICTOR UPDATE LOOP   ***                                  D3CSS     497
      DO 280 M=1,MAS                                                   D3CSS     498
      DO 270 N=2,NC                                                    D3CSS     499
      DO 270 I=1,4                                                     D3CSS     500
  270 CUP(I,N,M)=CU(I,N,M)+DZ*CUP(I,N,M)                               D3CSS     501
      CUP(1,1,M)=CU(1,1,M)+DZ*CUP(1,1,M)                               D3CSS     502
      IF (M .GT. ISWSMO)                                               D3CSS     503
     1   CUP(2,1,M)=CU(2,1,M)+DZ*CUP(2,1,M)                            D3CSS     504
      CUP(3,1,M)=CU(3,1,M)+DZ*CUP(3,1,M)                               D3CSS     505
  280 CONTINUE                                                         D3CSS     506
      IF (IDYAW .EQ. 1) GO TO 300                                      D3CSS     507
C   ***   SET SYMMETRY CONDITIONS   ***                                D3CSS     508
      DO 282 N=2,NC                                                    D3CSS     509
  282 CUP(4,N,1)=CUP(4,N,MC)=0.                                        D3CSS     510
      CUP(3,1,1)=CUP(3,1,MC)=CUP(3,NC,1)=CUP(3,NC,MC)=0.               D3CSS     511
C                                                                      D3CSS     512
C**********************************************************************D3CSS     513
C            *****   CORRECTOR-PREDICTOR LOOP   *****                  D3CSS     514
C        THIS LOOP SIMULTANEOUSLY CORRECTS ON THE LINE M-1,            D3CSS     515
C             COMPUTES Z-DIFFERENCES FOR NEW PREDICTOR ON LINE M-2,    D3CSS     516
C             AND FINDS THE CFL PARAMETER FOR THE NEXT DZ              D3CSS     517
C        J1,J2,J3 CORRESPONDS TO M-2,M-1,M RESPECTIVELY FOR            D3CSS     518
C             TRANG QUANTITIES.                                        D3CSS     519
C        K1,K2 CORRESPONDS TO M-2,M-1 RESPECTIVELY FOR                 D3CSS     520
C             TRANF QUANTITIES.                                        D3CSS     521
C        K1 CORRESPONDS TO M-2 FOR CF AND CE VECTORS                   D3CSS     522
C             BASED ON CORRECTED VALUES.                               D3CSS     523
C        K2 CORRESPONDS TO M-1 FOR CF AND CE (BASED ON                 D3CSS     524
C             CORRECTED AND/OR PREDICTED VALUES).                      D3CSS     525
C        J1,J2 CORRESPONDS TO M-2 FOR CG VECTOR BASED ON               D3CSS     526
C             CORRECTED AND PREDICTED VALUES RESPECTIVELY.             D3CSS     527
C        J3 CORRESPONDS TO M-1 FOR CG VECTOR BASED ON                  D3CSS     528
C             PREDICTED VALUES.                                        D3CSS     529
  300 J1=1 $ J2=2 $ J3=3 $ K1=1 $ K2=2                                 D3CSS     530
      M=MCM2S $ MR=0                                                   D3CSS     531
      CFL=0.0                                                          D3CSS     532
      TBZZ0=BZZ0(MA) $ TBZPHI0=BZPHI0(MA)                              D3CSS     533
      TBPP0=BPHPH0(MA)                                                 D3CSS     534
      DO 600 MT=1,MCP2                                                 D3CSS     535
```

```
      MM2=MB $ MB=M $ M=M+1                                                  D3CSS    536
      IF (M .NE. MC) GO TO 310 $ IF (MCM2S .EQ. 0) GO TO 310                 D3CSS    537
      M=1                                                                    D3CSS    538
  310 KK=K1 $ K1=K2 $ K2=KK                                                  D3CSS    539
      JJ=J1 $ J1=J2 $ J2=J3 $ J3=JJ $ IIC=IIP=4                              D3CSS    540
      IF (MT .LT. MCP) GO TO 320                                             D3CSS    541
      IF (M .NE. MCP) GO TO 315                                              D3CSS    542
      IIC=3 $ GO TO 345                                                      D3CSS    543
  315 IF (M .EQ. 1) GO TO 320                                                D3CSS    544
      IF (M .EQ. 2) GO TO 317                                                D3CSS    545
      IIP=3                                                                  D3CSS    546
C   ***   COMPUTE CG VECTOR AT Y=1+DY USING SYMMETRY CONDITIONS    ***       D3CSS    547
      CALL TRANG(YMCP,MCP,J2)                                                D3CSS    548
      CALL EVALSY(DUM,MA,J1,J2,J2,DUM)                                       D3CSS    549
      GO TO 560                                                              D3CSS    550
C   ***   COMPUTE CG VECTOR AT Y=1 (USING CORRECTED VALUES)    ***           D3CSS    551
  317 CALL EVALPR(DUM,1,DUM,J2,J2,DUM)                                       D3CSS    552
      GO TO 560                                                              D3CSS    553
  320 CALL TRANG(Y(M),M,J3)                                                  D3CSS    554
      PHI(M)=THETA*PHIO $ GY(M)=TG4(J3)                                      D3CSS    555
      IF (MT .GT. MC) GO TO 345                                              D3CSS    556
      IJUMP(M)=0 $ BZO(M)=BZ(M) $ BPHIO(M)=BPHI(M)                           D3CSS    557
      CALL BODY(M) $ CALL BODYP(M,J3)                                        D3CSS    558
      BZZO(M)=BZZT(J3) $ BZPHIO(M)=BZPHIT(J3) $ BPHPHO(M)=BPHPHT(J3)         D3CSS    559
      IF (MT .LT. MC) GO TO 325 $ IF (M .EQ. MC) GO TO 325                   D3CSS    560
      GO TO 345                                                              D3CSS    561
  325 COSPHI(M)=COS(PHI(M)) $ SINPHI(M)=SIN(PHI(M))                          D3CSS    562
      ICHECK=0                                                               D3CSS    563
      CALL DECODE(M,CUP,J3,NCMAX,MCMAX)                                      D3CSS    564
C   ***   CORRECTOR FOR SHOCK SHAPE    ***                                 D3CSS    565
      C(M)=CU(1,NC,M)=.5*(CU(1,NC,M)+CUP(1,NC,M)+DZ*(CZ(M)                   D3CSS    566
     1    -YZ(J3)*CPHI(M)/YPHI(J3)))                                         D3CSS    567
      IF (MT .GT. 3) GO TO 345                                               D3CSS    568
      IF (M .NE. 3) GO TO 330                                                D3CSS    569
      IIP=3 $ GO TO 345                                                      D3CSS    570
  330 IF (M .NE. 2) GO TO 600                                                D3CSS    571
      IF (MT .EQ. 2) GO TO 335                                               D3CSS    572
C   ***   COMPUTE CG VECTOR AT Y=-DY (I.E., Y=1-DY)    ***                   D3CSS    573
C   ***   NOTE THAT THE CALL TO TRANF IS TO OBTAIN R(N,MC-1) ONLY    ***     D3CSS    574
      CALL TRANF(MA,J1,K1)                                                   D3CSS    575
      CALL EVALPR(DUM,MA,DUM,J1,J2,DUM)                                      D3CSS    576
      BZ(MA)=BZO(MA) $ BPHI(MA)=BPHIO(MA)                                    D3CSS    577
      BZZO(MA)=TBZZO $ BZPHIO(MA)=TBZPHIO                                    D3CSS    578
      BPHPHO(MA)=TBPPO                                                       D3CSS    579
      GO TO 345                                                              D3CSS    580
  335 IIC=3                                                                  D3CSS    581
C   ***   COMPUTE CG VECTOR AT Y=-DY USING SYMMETRY CONDITIONS    ***       D3CSS    582
      CALL TRANG(Y0,0,J1)                                                    D3CSS    583
C   ***   THE FOLLOWING CALL IS TO FIND R(N,2) ONLY    ***                   D3CSS    584
      CALL TRANF(2,J3,K1)                                                    D3CSS    585
      CALL EVALSY(DUM,2,J2,J1,J2,DUM)                                        D3CSS    586
  345 CALL TRANF(MB,J2,K2)                                                   D3CSS    587
      CALL EVAL(1,MB,K2,J2,J3,K2)                                            D3CSS    588
      DO 400 N=2,NA $ NP=N+ICP                                               D3CSS    589
      DO 400 I=1,IIC                                                         D3CSS    590
      CU(I,N,MB)=.5*(CU(I,N,MB)+CUP(I,N,MB)                                  D3CSS    591
     1           -(CF(I,NP,K2)-CF(I,NP-1,K2))*DZDX                           D3CSS    592
```

```
     2          -(CG(I,N,J3)-CG(I,N,J2))*DZDY-D7*CE(I,N,K2))                    D3CSS     593
  400 CONTINUE                                                                  D3CSS     594
C    ***   CORRECTOR FOR WALL POINTS   ***                                      D3CSS     595
      CALL WALL(MB,J3,J2,J2,K2,1)                                               D3CSS     596
      CU(1,1,MB)=.5*(CU(1,1,MB)+CUP(1,1,MB)+D7*P7)                              D3CSS     597
      IF (MB .LE. ISWSMO) GO TO 420                                             D3CSS     598
      CU(2,1,MB)=.5*(CU(2,1,MB)+CUP(2,1,MB)+D7*S7)                              D3CSS     599
      GO TO 425                                                                 D3CSS     600
  420 CU(2,1,MB)=SZ                                                             D3CSS     601
  425 CONTINUE                                                                  D3CSS     602
      IF (IIC .EQ. 4) CU(3,1,MB)=.5*(CU(3,1,MB)+CUP(3,1,MB)+DZ*V2Z)            D3CSS     603
C  *** CORRECTOR FOR SHOCK POINTS  ***                                          D3CSS     604
      CALL SHOCK(MB,J3,J2,J2,K2,1)                                              D3CSS     605
      CU(2,NC,MB)=.5*(CU(2,NC,MB)+CUP(2,NC,MB)+DZ*DCZZ)                         D3CSS     606
      IF(IIC.EQ.4) CU(3,NC,MB)=.5*(CU(3,NC,MB)+CUP(3,NC,MB)+DZ*DCPHZ)           D3CSS     607
      DUM=CZ(MB)  $ DUMCP=CPHI(MB)                                              D3CSS     608
      ICHECK=1                                                                  D3CSS     609
      CALL DECODE(MB,CU,J2,NCMAX,MCMAX)                                         D3CSS     610
      BMB=B(MB) $  CMB=CZ(MB)      $   CMBB=C(MB)-BMB                           D3CSS     611
      DDCB=(DUM-CZMB)/CMBB                                                      D3CSS     612
      CPHIMB=CPHI(MB)    $ DDCPB=(DUMCP-CPHIMB)/CMBB                            D3CSS     613
      IF (IJUMP(MB) .EQ. 1) GO TO 435                                           D3CSS     614
      DDBP=DDBZ=0. $ GO TO 440                                                  D3CSS     615
  435 DUM1=BZ(MB) $ DUM2=BPHI(MB)                                               D3CSS     616
      BZZ=BZZT(J2) $ BPHPHI=BPHPHT(J2) $ BZPHI=BZPHIT(J2)                       D3CSS     617
      BZ(MB)=BZT(J2) $ BPHI(MB)=BPHIT(J2)                                       D3CSS     618
      CALL BODYPP(MB,J2)                                                        D3CSS     619
      DUMM=DUM2-BPHI(MB) $ DELBP=DUMM/B(MB)                                     D3CSS     620
      DDBP=DUMM/CMBB $ DELBZ=DUM1-BZ(MB) $ DDBZ=DELBZ/CMBB                      D3CSS     621
      CALL JUMP(DELBP,DELBZ,MB)                                                 D3CSS     622
      CU(2,1,MB)=SW(MB)                                                         D3CSS     623
      CU(1,1,MB)=ALOG(P(1,MB))                                                  D3CSS     624
      CU(3,1,MB)=V(1,MB)+BPHI(MB)*U(1,MB)/B(MB)                                 D3CSS     625
C    ***   UPDATE TRANF QUANTITIES USING CORRECTED CZ AND CPHI    ***           D3CSS     626
  440 DO 450 N=1,NC                                                             D3CSS     627
      XRN=XR(N,K2) $ DSX=R(N,MB)-BMB $ DSX1=DSX-CMBB                            D3CSS     628
      XZ(N,K2)=XZ(N,K2)+XRN*(DDCB*DSX-DDBZ*DSX1)                                D3CSS     629
      XPHI(N,K2)=XPHI(N,K2)-XRN*(DSX1*DDBP-DSX*DDCPB)                           D3CSS     630
      TF6(N,K2)=TF6(N,K2)+DDBZ-DDCB                                             D3CSS     631
      TF7(N,K2)=TF7(N,K2)+DDBP-DDCPB                                            D3CSS     632
  450 CONTINUE                                                                  D3CSS     633
      CALL EVAL(0,MB,K2,J2,J2,K2)                                               D3CSS     634
C    ***   COMPUTE Z DIFFERENCES FOR PREDICTOR              ***                 D3CSS     635
C    ***   NOTE THAT CP AND CUP STORE THESE QUANTITIES      ***                 D3CSS     636
      IF (M .EQ. 2) GO TO 600                                                   D3CSS     637
  560 DO 580 N=2,NA $ NP=N+IPC                                                  D3CSS     638
      DO 580 I=1,IIP                                                            D3CSS     639
      CUP(I,N,MM2)=-(CF(I,NP,K1)-CF(I,NP-1,K1))*DDX                             D3CSS     640
     1            -(CG(I,N,J2)-CG(I,N,J1))*DDY-CE(I,N,K1)                      D3CSS     641
  580 CONTINUE                                                                  D3CSS     642
      IF (MOD10N .EQ. -1) MOD1=1                                                D3CSS     643
      CALL WALL(MM2,J2,J1,J1,K1,0)                                              D3CSS     644
      IF (MOD10N .EQ. -1) MOD1=0                                                D3CSS     645
      CUP(1,1,MM2)=PZ $ CUP(3,1,MM2)=V2Z                                        D3CSS     646
      CUP(2,1,MM2)=SZ                                                           D3CSS     647
      CALL SHOCK(MM2,J2,J1,J1,K1,0)                                             D3CSS     648
      CUP(1,NC,MM2)=DCZ  $ CUP(2,NC,MM2)=DCZZ  $ CUP(3,NC,MM2)=DCPHZ            D3CSS     649
```

```
  600 CONTINUE                                                          D3CSS     650
      IF (ISWDIF .EQ. 0) GO TO 604                                      D3CSS     651
      IPC=1-IPC $ ICP=IPC $ BJ=-BJ                                      D3CSS     652
  604 CONTINUE                                                          D3CSS     653
C*********************************************************************** D3CSS     654
C                                                                       D3CSS     655
C   ***   OUTPUT  -  FORCES  -  MOMENTS   ***                           D3CSS     656
      IF (MCM2S .EQ. 0) GO TO 610                                       D3CSS     657
      PHI(MC)=PHIO $ C(MC)=C(1) $ CZ(MC)=CZ(1) $ CPHI(MC)=CPHI(1)       D3CSS     658
      DO 605 N=1,NC                                                     D3CSS     659
      R(N,MC)=R(N,1) $ U(N,MC)=U(N,1) $ V(N,MC)=V(N,1)                  D3CSS     660
      W(N,MC)=W(N,1) $ P(N,MC)=P(N,1) $ D(N,MC)=D(N,1)                  D3CSS     661
  605 CONTINUE                                                          D3CSS     662
  610 IF (IPRINT .EQ. 1) GO TO 675                                      D3CSS     663
      IF ((Z-PRINTZ) .GT. DZPRINT) GO TO 670                            D3CSS     664
      DO 625 I=1,5                                                      D3CSS     665
      IF (Z .LT. ZPRINT(I)) GO TO 650                                   D3CSS     666
  625 CONTINUE                                                          D3CSS     667
      GO TO 700                                                         D3CSS     668
  650 IF (MOD(K,KOUT(I)) .NE. 0) GO TO 700                              D3CSS     669
  670 IPRINT=1                                                          D3CSS     670
  675 CALL FIELD $ PRINTZ=Z                                             D3CSS     671
C                                                                       D3CSS     672
C      ACH IS THE MACH NUMBER                                           D3CSS     673
C      ATTA IS THE ANGLE OF ATTACK                                      D3CSS     674
C      CONE IS THE SHOULDER ANGLE                                       D3CSS     675
C      K IS THE STATION NUMBER                                          D3CSS     676
C      NC IS THE NUMBER OF POINTS IN RADIAL DIRECTION                   D3CSS     677
C      MC IS THE NUMBER OF PLANES IN TANGENTIAL DIRECTION               D3CSS     678
C      Z IS THE LENGTH ALONG BODY AXIS                                  D3CSS     679
C      PHI IS THE ANGLE OF PLANE IN TANGENTIAL DIRECTION                D3CSS     680
C      CZ IS +TAN(SIGMA)                                                D3CSS     681
C      CPHI IS -TAN(DELTA)                                              D3CSS     682
C      INDEX  1 MEANS FLOW VARIABLES ON THE BODY                        D3CSS     683
C      INDEX NC MEANS FLOW VARIABLES ON THE SHOCK                       D3CSS     684
C                                                                       D3CSS     685
  700 CALL INTEG(1)                                                     D3CSS     686
  725 WRITE (16) NC,MC,ATTA,YAW,ACH,GAMMA,PINF,DINF,PHIO,K,Z            D3CSS     687
     A  ,NGAS,NTEST,RRX                                                 D3CSS     688
     1  ,FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ                     D3CSS     689
     2    ,(PHI(M),C(M),CZ(M),CPHI(M),M=1,MC)                           D3CSS     690
     3    ,(((R(N,M),U(N,M),V(N,M),W(N,M),P(N,M),D(N,M),M=1,MC),N=1,NC) D3CSS     691
  750 IF (Z .LT. ZEND .AND. K .LT. KA) GO TO 200                        D3CSS     692
      IF (IPRINT .EQ. 0) CALL FIELD                                     D3CSS     693
      WRITE (17) NC,MC,ATTA,YAW,ACH,GAMMA,PINF,DINF,PHIO,K,Z            D3CSS     694
     A  ,NGAS,NTEST,RRX                                                 D3CSS     695
     1  ,FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ                     D3CSS     696
     2    ,(PHI(M),C(M),CZ(M),CPHI(M),M=1,MC)                           D3CSS     697
     3    ,(((R(N,M),U(N,M),V(N,M),W(N,M),P(N,M),D(N,M),M=1,MC),N=1,NC) D3CSS     698
      CALL OUT $ STOP $ END                                             D3CSS     699
```

```
      SUBROUTINE BODYP(M,J3)                                                BODYP     2
C                                                                           BODYP     3
C      BODYP COMPUTES CERTAIN BODY PARAMETERS NEEDED IN                     BODYP     4
C      SUBROUTINE WALL.  THIS ROUTINE ALSO TESTS FOR DISCONTINUITIES        BODYP     5
C      IN BZ AND BPHI.                                                      BODYP     6
C      J3=1,2,3 IS A LINE INDEX FOR BODY PARAMETERS                         BODYP     7
C                                                                           BODYP     8
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                            NEWCOM    1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                   NEWCOM    2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)        NEWCOM    3
      COMMON CU(4,20,25),CUP(4,20,25)                                       NEWCOM    4
C   ***   END OF BLANK COMMON   ***                                         CD3CSS   32
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                          CBODY     2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)              CBODY     3
      COMMON /CBODYP/ DZ,PHI1J,PHI2J,BZT(3),BPHIT(3),BZZT(3)                CBODYP    2
     1   ,BPHPHT(3),BZPHIT(3),BZO(25),BPHIO(25),BPHPHO(25),BZPHIO(25)      CBODYP    3
     2   ,BZZO(25)                                                         CBODYP    4
      COMMON /BLK03/ ICFL,A2(3),A3(3),A4(3),A5(3),A7(3)                     BLK03     2
     1   ,IJUMP(25),IJUMP1(25),IJMPKT(25)                                  BLK03     3
C                                                                           BODYP    10
      BZM=BZT(J3)=BZ(M) $ BPHIT(J3)=BPHI(M)                                 BODYP    11
      BPHPHT(J3)=BPHPHI $ BZPHIT(J3)=BZPHI $ BZZT(J3)=BZZ                   BODYP    12
      BZOM=BZO(M) $ BZPHOM=BZPHIO(M)                                        BODYP    13
      TEST1=AMAX1(ABS(BZZ),ABS(BZZO(M))) $ TEST1=ABS(BZM-BZOM)-DZ*TEST1     BODYP    14
      TEST2=AMAX1(ABS(BZPHI),ABS(BZPHOM))                                   BODYP    15
      TEST2=ABS(BPHI(M)-BPHIO(M))-DZ*TEST2                                  BODYP    16
      IF (TEST1 .LE. 1.E-6 .AND. TEST2 .LE. 1.E-6) GO TO 200                BODYP    17
C   ***   IF (PHI1J .LT. PHI(M) .LT. PHI2J) THEN NO JUMP   ***              BODYP    18
      IF (PHI(M) .GE. PHI2J) GO TO 10                                       BODYP    19
      IF (PHI(M) .LE. PHI1J) GO TO 10                                       BODYP    20
      ICFL=1 $ IJUMP1(M)=2 $ GO TO 200                                      BODYP    21
   10 BZ(M)=BZOM $ BPHI(M)=BPHIO(M) $ IJUMP(M)=1                            BODYP    22
      BPHPHI=BPHPHO(M) $ BZPHI=BZPHOM $ BZZ=BZZO(M)                         BODYP    23
      ENTRY BODYPP                                                          BODYP    24
  200 BM=B(M) $ BZM=BZ(M) $ PHIZ=-YZ(J3)/YPHI(J3)                           BODYP    25
      BPOB=BPHI(M)/BM $ BPOB2=BPOB**2 $ DUM=1.+BZM**2                       BODYP    26
      A22=DUM+BPOB2 $ DUM1=(BPHPHI/BM-BPOB2)                                BODYP    27
      DUM2=(BZZ+BZPHI*PHIZ)/DUM $ DUM3=BZPHI/BM                             BODYP    28
      DUM4=DUM3-BZM*BPOB/BM                                                 BODYP    29
      A2(J3)=SQRT(A22) $ A3(J3)=DUM1/YPHI(J3)                               BODYP    30
      A4(J3)=DUM4+PHIZ*DUM1 $ A5(J3)=BZPHI/YPHI(J3)                         BODYP    31
      A7(J3)=DUM2*DUM                                                       BODYP    32
      RETURN                                                                BODYP    33
      END                                                                   BODYP    34
```

```
      SUBROUTINE DECODE(M,CV,J,NDIM,MDIM)                                    DECODE     2
C                                                                            DECODE     3
C       DECODE FINDS FLOW VARIABLES FROM THE CONSERVATION VECTOR, CV.        DECODE     4
C       THIS VERSION ASSUMES THAT CV(1,1,M)=LOG(P), CV(2,1,M)=S,             DECODE     5
C       CV(3,1,M)=V2=V+(BPHI/R)*U                                            DECODE     6
C       CV(1,NC,M)=C    CV(2,NC,M)=CZ    CV(3,NC,M)=CPHI                     DECODE     7
C       THIS ROUTINE CONTAINS SELECTIVE SMOOTHING OF CONSERVATION            DECODE     8
C       VECTOR WHEN DECODED PRESSURE IS NEGATIVE.                            DECODE     9
C       NOTE THAT CV IS EITHER CU OR CUP AS THE                              DECODE    10
C       SEQUENCE IN THE MAIN PROGRAM INDICATES                               DECODE    11
C       J=1,2,3 IS A LINE INDEX FOR BODYP PARAMETERS                         DECODE    12
C                                                                            DECODE    13
      COMMON/RGASS/AX,HX,TX,RRX,GX,NTEST,NGAS,NFIRST                         DECODE    14
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                             NEWCOM     1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                    NEWCOM     2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)         NEWCOM     3
      COMMON CU(4,20,25),CUP(4,20,25)                                        NEWCOM     4
C   ***   END OF BLANK COMMON   ***                                          CD3CSS    32
      COMMON /CDECODE/ GFF,GG,GIM1,HINF,V1INF,V2INF,WINX,DINF2               CDECODE    2
     1   ,ICHECK,X1K1NEG,X1K1NP2,GC(20,25)                                   CDECODE    3
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                           CBODY      2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)                CBODY      3
      COMMON /BLK02/ THETA,DY,TG4(3),TG5(3),TG6(3)                           BLK02      2
     1   ,X(20),XZ(20,2),XR(20,2),XPHI(20,2),Y(25)                           BLK02      3
     2   ,TF4(20,2),TF6(20,2),TF7(20,2)                                      BLK02      4
      COMMON /BLK03/ ICFL,A2(3),A3(3),A4(3),A5(3),A7(3)                      BLK03      2
     1   ,IJUMP(25),IJUMP1(25),IJMPKT(25)                                    BLK03      3
      COMMON /BLK04/ GAMMA,GB,GD,GE,GA2,DDX,DDY,HOT2,ELIM,LCNT,ISWSMO,NA     BLK04      2
     1   ,SW(25),GM(20,25)                                                   BLK04      3
C                                                                            DECODE    16
      DIMENSION CV(4,NDIM,MDIM)                                              DECODE    17
      U3=CV(3,1,M)                                                           DECODE    18
      SW(M)=SWM=CV(2,1,M)                                                    DECODE    19
      T1=BZ(M) $ T2=BPHI(M)/R(M) $ T3=1.+T2**2                               DECODE    20
      IF (CV(1,1,M) .GT. -600. .AND. CV(1,1,M) .LT. 700.) GO TO 2001         DUMP       5
      WRITE (6,3456) M,CV(1,1,M)                                             DUMP       6
 3456 FORMAT(1H1,*IN SUBROUTINE DECODE THE LOG OF PRESSURE ON PLANE*,I3,     DUMP       7
     1   *  ON THE BODY IS*,1PE15.6,5X,*---  STOP  ---*)                     DUMP       8
      CALL SAVE(DUM,DUM,DUM)                                                 DUMP       9
 2001 CONTINUE                                                               DUMP      10
      P(1,M)=PM=EXP(CV(1,1,M))                                               DECODE    21
      CALL RGAS(PM,DM,SWM,5)                                                 DECODE    22
      D(1,M)=DM                                                              DECODE    23
      ASQ(1,M)=AX*AX                                                         DECODE    24
      QSQ=HOT2-2.*HX                                                         DECODE    25
      CDUMP=QSQ*T3-U3*U3                                                     DUMP      11
      IF (CDUMP .GE. 0.) GO TO 1001                                          DUMP      12
      CALL DMPSQRT(6HDECODE,1,Z,K,M,1,CDUMP)                                 DUMP      13
 1001 WM=SQRT(CDUMP)/A2(J)                                                   DUMP      14
      IF(NTEST.GE.0)GO TO 18                                                 DECODE    27
      GMT=1./(1.-PM/(DM*HX))                                                 DECODE    28
      GM(1,M)=GMT                                                            DECODE    29
   18 W(1,M)=WM                                                              DECODE    30
      V(1,M)=VM=(U3-T2*T1*WM)/T3                                             DECODE    31
      U(1,M)=UM=T1*WM+T2*VM                                                  DECODE    32
      DO 10 N=2,NA                                                           DECODE    33
      KNT=0                                                                  DECODE    34
```

```
      IF(NTEST)19,20,20                                                 DECODE    35
C.....USE LAST VALUE OF GAMMA AS ESTIMATE FOR NEW VALUE                 DECODE    36
   19 ICNT=0                                                            DECODE    37
      GMT=GM(N,M)                                                       DECODE    38
   17 GG=GMT*GMT-1.                                                     DECODE    39
      GFF=GMT+1.                                                        DECODE    40
   20 CONTINUE                                                          DECODE    41
      U1=CV(1,N,M)   $U2=CV(2,N,M)   $U21= U2/U1                       DECODE    42
      UNM=U(N,M)=CV(3,N,M)/U1   $VNM=V(N,M)=CV(4,N,M)/U1               DECODE    43
      VK=UNM**2+VNM**2                                                  DECODE    44
      PHT=HOTP-VK                                                       DECODE    45
      CRAP1=U21*U21                                                     DECODE    46
      PHTU=PHT-CRAP1                                                    DECODE    47
      IF(KNT.LT.2)GO TO 32                                              DECODE    48
      X1K1NEG=X1K1S $ X1K1NP2=X1K1SP2                                   DECODE    49
   32 IF(PHTU.GT.0.0)GO TO 91                                           DECODE    50
      IF(N.EQ.NA) WRITE(6,3268)Z,N,M,K                                  DECODE    51
      IF(N.EQ.NA)GO TO 91                                               DECODE    52
 3268 FORMAT(1H0,*P IS NEGATIVE BUT NOT AVERAGED *,5X,*Z,N,M,K=*,F7.3,  DECODE    53
     1  3I5)                                                            DECODE    54
      KNT=KNT+1 $ IF(KNT.LE.2)GO TO 34                                  DECODE    55
      WRITE(6,3267)Z,N,M,K                                              DECODE    56
 3267 FORMAT(1H0,*P STAYS NEGATIVE AFTER AVERAGING*,5X,*Z,N,M,K =*,      DECODE    57
     1    F7.3,3I5)                                                    DECODE    58
      IF (NTEST .LT. 0) CALL SAVE(DUM,DUM,DUM)                          DUMP      15
      GO TO 91                                                          DECODE    59
   34 IF(KNT.EQ.1)GO TO 4                                               DECODE    60
      X1K1S=X1K1NEG $ X1K1SP2=X1K1NP2                                   DECODE    61
      X1K1NEG=0. $ X1K1NP2=2.                                           DECODE    62
    4 IF(N.EQ.2)GO TO 5                                                 DECODE    63
      DO 6 I=1,4                                                        DECODE    64
    6 CV(I,N,M )=(CV(I,N+1,M)+X1K1NEG*CV(I,N,M)+CV(I,N-1,M))/X1K1NP2    DECODE    65
      GO TO 2                                                           DECODE    66
    5 TCU=DM*WM                                                         DECODE    67
      CV(1,2,M)=(CV(1,3,M)+X1K1NEG*CV(1,2,M)+TCU)/X1K1NP2               DECODE    68
      CV(2,2,M)=(CV(2,3,M)+X1K1NEG*CV(2,2,M)+PM+TCU*WM)/X1K1NP2         DECODE    69
      CV(3,2,M)=(CV(3,3,M)+X1K1NEG*CV(3,2,M)+UM*TCU)/X1K1NP2            DECODE    70
      CV(4,2,M)=(CV(4,3,M)+X1K1NEG*CV(4,2,M)+VM*TCU)/X1K1NP2            DECODE    71
    2 WRITE(6,3273) X1K1NEG,Z,N,M,K                                     DECODE    72
 3273 FORMAT(1H0,*THE CONSERVATION VECTOR IS 1-*,F3.0,*-1*,5X,          DECODE    73
     1       *Z,N,M,K = *,F7.3,3I5)                                     DECODE    74
      GO TO 20                                                          DECODE    75
   91 CONTINUE                                                          DECODE    76
      CRAPP=PHTU/CRAP1                                                  DECODE    77
      CRAP2=CRAPP*GG                                                    DECODE    78
      CDUMP=1.-CRAP2                                                    DUMP      16
      IF (CDUMP .GE. 0.) GO TO 1002                                     DUMP      17
      CALL DMPSQRT(6HDECODE,2,Z,K,M,N,CDUMP)                            DUMP      18
 1002 CRAP3=CRAP2/((1.+SQRT(CDUMP))*GFF)                                DUMP      19
      WNM=(1.-CRAP3)*U21                                                DECODE    80
      PNM=P(N,M)=CRAP3*U2                                               DECODE    81
      DNM=D(N,M)=U1/WNM                                                 DECODE    82
      IF(NTEST)21,22,22                                                 DECODE    83
   21 CALL HRGAS(PNM,DNM,DUMMY,2)                                       DECODE    84
      ERR=2.*HX+WNM*WNM-PHT                                             DECODE    85
      WNM2=WNM                                                          DECODE    86
      IF(ABS(ERR/HX)-ELIM)23,23,24                                      DECODE    87
```

```
   24 IF(ICNT)28,16                                                    DECODE     88
C.....QUASIREAL GAS APPROACH FAILS.ITERATE FOR SOLUTION.               DECODE     89
C.....FIRST BRACKET CORRECT SOLUTION                                   DECODE     90
   16 WNMX=WNM1=WNM                                                    DECODE     91
      ERRX=ERR1=ERR                                                    DECODE     92
      GCC=GC(N,M)                                                      DECODE     93
      WNM=ERR/GCC+WNM                                                  DECODE     94
      IF(GCC.LT.1.E+99)GO TO 80                                        DECODE     95
      GMT=1./(1.-PNM/(DNM*HX))                                         DECODE     96
      ICNT=ICNT+1                                                      DECODE     97
      GO TO 17                                                         DECODE     98
   25 ICNT=ICNT+1                                                      DECODE     99
   80 IF(WNM.GT.U21)WNM=U21-1.E-10                                     DECODE    100
      PNM=U2-WNM*U1                                                    DECODE    101
      DNM=U1/WNM                                                       DECODE    102
      CALL HRGAS(PNM,DNM,DUMMY,2)                                      DECODE    103
      ERR2=ERR=2.*HX*WNM*WNM-PHT                                       DECODE    104
      WNM2=WNM                                                         DECODE    105
      IF(ABS(ERR/HX)-ELIM )23,23,26                                    DECODE    106
C.....SOLUTION NOT ACCURATE ENOUGH                                     DECODE    107
   26 IF(ICNT-LCNT)28,28,30                                            DECODE    108
   30 WRITE(6,2000)N,M                                                 DECODE    109
 2000 FORMAT(1H1,2I5,*ITERATION LIMIT EXCEED*)                         DECODE    110
      STOP                                                             DECODE    111
   28 IF(ERR*ERR1)27,29,29                                             DECODE    112
C.....SOLUTION NOT BRACKETED                                           DECODE    113
C.....EXTRAPOLATE FOR NEW SOLUTION                                     DECODE    114
   29 DWNM=(WNM-WNM1)*ERR1/(ERR1-ERR)                                  DECODE    115
      IF(ABS(DWNM).GT..05*ABS(WNM))DWNM=.05*ABS(WNM)*ABS(DWNM)/DWNM    DECODE    116
      WNM2=WNM1+DWNM                                                   DECODE    117
      IF(ABS(ERR).LT.ABS(ERR1))GO TO 79                                DECODE    118
      WNM=WNM2                                                         DECODE    119
      GO TO 25                                                         DECODE    120
   79 ERR1=ERR                                                         DECODE    121
      WNM1=WNM                                                         DECODE    122
      WNM=WNM2                                                         DECODE    123
      GO TO 25                                                         DECODE    124
C.....SOLUTION IS BRACKETED                                            DECODE    125
   27 ICNT=ICNT+1                                                      DECODE    126
      WNM2=WNM1+(WNM-WNM1)*ERR1/(ERR1-ERR)                             DECODE    127
      PNM=U2-WNM2*U1                                                   DECODE    128
      DNM=U1/WNM2                                                      DECODE    129
      CALL HRGAS(PNM,DNM,DUMMY,2)                                      DECODE    130
      ERR2=2.*HX+WNM2*WNM2-PHT                                         DECODE    131
      IF(ABS(ERR2/HX)-ELIM)23,23,42                                    DECODE    132
   42 IF(ICNT-LCNT)43,43,30                                            DECODE    133
   43 IF(ERR2*ERR)44,44,45                                             DECODE    134
C.....WNM2 AND WNM BRACKET SOLUTION                                    DECODE    135
   44 WNM1=WNM2                                                        DECODE    136
      ERR1=ERR2                                                        DECODE    137
      GO TO 27                                                         DECODE    138
C.....WNM1 AND WNM2 ARE BRACKET POINTS                                 DECODE    139
   45 WNM=WNM2                                                         DECODE    140
      ERR=ERR2                                                         DECODE    141
      GO TO 27                                                         DECODE    142
C.....CONVERGENCE ACHIEVED                                             DECODE    143
   23 GM(N,M)=GMT=1./(1.-PNM/(DNM*HX))                                 DECODE    144
```

```
      IF (ICNT .GT. 1) GC(N,M)=(ERR2-ERRX)/(WNMX-WNM2)                  DECODE   145
      W(N,M)=WNM2                                                       DECODE   146
      P(N,M)=PNM                                                        DECODE   147
      D(N,M)=DNM                                                        DECODE   148
      IF(ICHECK.EQ.1)CALL ARGAS(PNM,DNM,AX2,1)                          DECODE   149
      GO TO 10                                                          DECODE   150
   22 CONTINUE                                                          DECODE   151
      AX2=GAMMA*PNM/DNM                                                 DECODE   152
      W(N,M)=WNM                                                        DECODE   153
   10 ASQ(N,M)=AX2                                                      DECODE   154
C                                                                       DECODE   155
C     SHOCK SHAPE AND THEREFORE V FREE STREAM ARE KNOWN.                DECODE   156
C     PRESSURE AND DENSITY ARE CALCULATED FROM SHOCK RELATIONS.         DECODE   157
C                                                                       DECODE   158
      UINX=-(V1INF*COSPHI(M)+V2INF*SINPHI(M))                           DECODE   159
      VINX=V1INF*SINPHI(M)-V2INF*COSPHI(M)                              DECODE   160
      CM=C(M)=CV(1,NC,M)   $ CZM=CZ(M)=CV(2,NC,M)                       DECODE   161
      CPHIM=CPHI(M)=CV(3,NC,M)   $ CPHC=CPHIM/CM                        DECODE   162
      DMU2=CZM**2+CPHC**2+1.                                            DECODE   163
      CVNMU=WINX*CZM-UINX*VINX*CPHC                                     DECODE   164
      CVN2=CVNMU**2/DMU2                                                DECODE   165
      IF(NTEST.GE.0)GO TO 48                                            DECODE   166
      ICNT=0                                                            DECODE   167
      GMT=GM(NC,M)                                                      DECODE   168
   85 GE=(GMT+1.)*.5                                                    DECODE   169
      GD=(GMT-1.)*.5                                                    DECODE   170
      GAMMA=GMT                                                         DECODE   171
C.....CONSTANT GAMMA FORMULAS                                           DECODE   172
   48 CDUMP=CVN2*(CVN2*DINF2+PINF*DINF*(2.+8.*GE*GD/GIM1))              DUMP      20
     1      +PINF*PINF*GAMMA*GAMMA                                      DUMP      21
      IF (CDUMP .GE. 0.) GO TO 1003                                     DUMP      22
      CALL DMPSQRT(6HDECODE,3,Z,K,M,NC,CDUMP)                           DUMP      23
 1003 PNM=(PINF+DINF*CVN2+SQRT(CDUMP))/(2.*GE)                          DUMP      24
      DNM=DINF*CVN2/(CVN2+(PINF-PNM)/DINF)                              DECODE   175
      AX2=GAMMA*PNM/DNM                                                 DECODE   176
      IF(NTEST.GT.50)50                                                 DECODE   177
C     REAL GAS                                                          DECODE   178
   51 CALL ARGAS(PNM,DNM,DUMMY,2)                                       DECODE   179
      SN2=CVN2*DINF2/(DNM*DNM)                                          DECODE   180
      DCH=3.*SN2                                                        DECODE   181
      ERR=(HX-HINF)*2.+SN2-CVN2                                         DECODE   182
      IF(ABS(ERR/HX)-ELIM)52,52,53                                      DECODE   183
C.....NOT ACCURATE ENOUGH                                               DECODE   184
   53 IF(ICNT)55,81                                                     DECODE   185
   81 GMT=1./(1.-PNM/(DNM*HX))                                          DECODE   186
      ICNT=ICNT+1                                                       DECODE   187
      CVTN=SN2                                                          DECODE   188
      ERR1=ERR                                                          DECODE   189
      GO TO 85                                                          DECODE   190
   54 ICNT=ICNT+1                                                       DECODE   191
      CDUMP=CVN2/CVTN                                                   DUMP      25
      IF (CDUMP .GE. 0.) GO TO 1004                                     DUMP      26
      CALL DMPSQRT(6HDECODE,4,Z,K,M,NC,CDUMP)                           DUMP      27
 1004 DNM=DINF*SQRT(CDUMP)                                              DUMP      28
      PNM=PINF+DINF*CVN2*(1.-DINF/DNM)                                  DECODE   193
      CALL ARGAS(PNM,DNM,DUMMY,2)                                       DECODE   194
      ERR1=2.*(HX-HINF)+CVTN-CVN2                                       DECODE   195
```

```
      IF(ABS(ERR1/HX)-ELIM)52,52,58                                    DECODE    196
   58 IF(ICNT-LCNT)55,55,57                                            DECODE    197
   57 WRITE(6,2003)N                                                   DECODE    198
      STOP                                                             DECODE    199
 2003 FORMAT(1H1,*LIMIT ON SHOCK ITERATIONS EXCFEDED*,I5)              DECODE    200
   55 IF(ERR1*ERR)56,56,76                                             DECODE    201
C.....POINT NOT BRACKETED                                             DECODE    202
   76 DCVN=(SN2-CVTN)*ERR1/(ERR1-ERR)                                  DECODE    203
      IF(ABS(DCVN).GT.ABS(3.*DCH))DCVN=3.*DCH*ABS(DCVN)/DCVN           DECODE    204
      CVT2=CVTN+DCVN                                                   DECODE    205
      IF(ABS(ERR).GT.ABS(ERR1))GO TO 78                                DECODE    206
      SN2=CVT2                                                         DECODE    207
      GO TO 54                                                         DECODE    208
   78 SN2=CVTN                                                         DECODE    209
      ERR=ERR1                                                         DECODE    210
      CVTN=CVT2                                                        DECODE    211
      GO TO 54                                                         DECODE    212
C.....SOLUTION BRACKETED                                              DECODE    213
   56 CVT2=SN2+(CVTN-SN2)*ERR/(ERR-ERR1)                               DECODE    214
      ICNT=ICNT+1                                                      DECODE    215
      CDUMP=CVN2/CVT2                                                  DUMP       29
      IF (CDUMP .GE. 0.) GO TO 1005                                    DUMP       30
      CALL DMPSQRT(6HDECODE,5,Z,K,M,NC,CDUMP)                          DUMP       31
 1005 DNM=DINF*SQRT(CDUMP)                                             DUMP       32
      PNM=PINF+DINF*CVN2*(1.-DINF/DNM)                                 DECODE    217
      CALL HRGAS(PNM,DNM,DUMMY,2)                                      DECODE    218
      ERRN=2.*(HX-HINF)+CVT2-CVN2                                      DECODE    219
      IF(ABS(ERRN/HX)-ELIM)52,52,61                                    DECODE    220
   61 IF(ICNT-LCNT)62,62,58                                            DECODE    221
   62 IF(ERRN*ERR)68,69,69                                             DECODE    222
C.....ERR AND ERRN BRACKET THE SOLUTION                               DECODE    223
   68 CVTN=CVT2                                                        DECODE    224
      ERR1=ERRN                                                        DECODE    225
      GO TO 56                                                         DECODE    226
C.....ERR1 AND ERRN BRACKET THE SOLUTION                              DECODE    227
   69 SN2=CVT2                                                         DECODE    228
      ERR=ERRN                                                         DECODE    229
      GO TO 56                                                         DECODE    230
C.....CONVERGENCE ACHIEVED                                            DECODE    231
   52 GM(NC,M)=1./(1.-PNM/(DNM*HX))                                    DECODE    232
      CALL ARGAS(PNM,DNM,AX2,1)                                        DECODE    233
   50 CONTINUE                                                         DECODE    234
      XLCA=CVNMU*(1.-DINF/DNM)/DMU2                                    DECODE    235
      UNM=U(NC,M)=UINX+XLCA  $  VNM=V(NC,M)=VINX-XLCA*CPHC             DECODE    236
      WNM=W(NC,M)=WINX-XLCA*CZM                                        DECODE    237
      P(NC,M)=PNM                                                      DECODE    238
      D(NC,M)=DNM                                                      DECODE    239
      ASQ(NC,M)=AX2                                                    DECODE    240
      RETURN                                                           DECODE    241
      END                                                              DECODE    242
```

```
      SUBROUTINE EVAL(L,M,IT,JSG,JCG,JCFCE)                                  EVAL        2
C                                                                            EVAL        3
C      EVAL COMPUTES FLUX AND SOURCE VECTORS CF,CG,CF                        EVAL        4
C      FROM FLOW VARIABLES, TRANG AND TRANF QUANTITIES.                      EVAL        5
C      EVAL ALSO COMPUTES CFL PARAMETER.                                     EVAL        6
C      IF L=0 EVALUATE FUNCTIONS AND CHECK CFL (PREDICTOR)                   EVAL        7
C      IF L=1 EVALUATE FUNCTIONS (CORRECTOR)                                 EVAL        8
C      IT=1,2    IS A LINE INDEX FOR XZ,XR,XPHI,TF4,TF5,TF6,TF7              EVAL        9
C      JSG=1,2,3 IS A LINE INDEX FOR PHI,YPHI,Y7,TG4,TG5,TG6                 EVAL       10
C      JCG,JCFCE ARE LINE INDEXES FOR CG AND (CF,CE) RESPECTIVELY            EVAL       11
C                                                                            EVAL       12
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                             NEWCOM      1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                    NEWCOM      2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)         NEWCOM      3
      COMMON CU(4,20,25),CUP(4,20,25)                                        NEWCOM      4
C   ***   END OF BLANK COMMON   ***                                          CD3CSS     32
      COMMON /CEVAL/ NCFL,JCFL,MCFL,CFL,RJ                                   CEVAL       2
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                           CBODY       2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)                CBODY       3
      COMMON /BLK01/ CZY(3),CPHIY(3),V2(3),VOWY(3),PWY(3),SWY(3)             BLK01       2
     1   ,UNOR(3,2),CF(4,20,2),CG(4,20,3),CE(4,20,2)                         BLK01       3
      COMMON /BLK02/ THETA,DY,TG4(3),TG5(3),TG6(3)                           BLK02       2
     1   ,X(20),XZ(20,2),XR(20,2),XPHI(20,2),Y(25)                           BLK02       3
     2   ,TF4(20,2),TF6(20,2),TF7(20,2)                                      BLK02       4
      COMMON /BLK04/ GAMMA,GB,GD,GE,GA2,DDX,DDY,HOT2,ELIM,LCNT,ISWSMO,NA     BLK04       2
     1   ,SW(25),GM(20,25)                                                   BLK04       3
      CZY(JCG)=CZ(M) $ CPHIY(JCG)=CPHI(M)                                    EVAL       14
C                                                                            EVAL       15
      YPHIJ=YPHI(JSG) $ YZJ=YZ(JSG) $ TG5J=TG5(JSG)                          EVAL       16
      DO 10 N=1,NC                                                           EVAL       17
      UNM=U(N,M)   $  WNM=W(N,M)   $  PNM=P(N,M)   $  VNM=V(N,M)            EVAL       18
      DNM=D(N,M)   $  RNM=R(N,M)   $ XZN=XZ(N,IT)   $  XRN=XR(N,IT)         EVAL       19
      XPHIN=XPHI(N,IT)   $  TF4N=TF4(N,IT)   $  TF6N=TF6(N,IT)              EVAL       20
      TF7N=TF7(N,IT)                                                         EVAL       21
      YPOR=YPHIJ/RNM   $XPOR=XPHIN/RNM                                       EVAL       22
      VOR=VNM/RNM $ AD=XRN*UNM+XPHIN*VOR $ AA=WNM*XZN+AD                     EVAL       23
      IF (N .LE. 3) UNOR(N,JCFCE)=AA                                         EVAL       24
      BB=WNM*YZJ+YPHIJ*VOR                                                   EVAL       25
      XRP=XRN*PNM                                                            EVAL       26
      IF (N .NE. 1) GO TO 4                                                  EVAL       27
      PWY(JCG)=PNM $ VOWY(JCG)=VNM/WNM $ SWY(JCG)=SW(M)                      EVAL       28
      V2(JCG)=VNM+APHI(M)*UNM/B(M)                                           EVAL       29
    4 CF(1,N,JCFCE)=RA=DNM*AA                                                EVAL       30
      CF(2,N,JCFCE)=XZN*PNM+RA*WNM                                           EVAL       31
      CF(3,N,JCFCE)=XRP+RA*UNM                                               EVAL       32
      CF(4,N,JCFCE)=XPOR*PNM+RA*VNM                                          EVAL       33
      CG(1,N,JCG) = RA=DNM*BB                                                EVAL       34
      CG(2,N,JCG) = YZJ*PNM+RA*WNM                                           EVAL       35
      CG(3,N,JCG) = RB*UNM                                                   EVAL       36
      CG(4,N,JCG) = YPOR*PNM+RB*VNM                                          EVAL       37
      CE(1,N,JCFCE)=E1=DNM*(UNM/RNM+TF4N*AA+TG5J*BB+TF6N*WNM                 EVAL       38
     1  +TF7N*VOR)                                                           EVAL       39
      CE(2,N,JCFCE)=WNM*E1+(TG5J*YZJ+TF4N*XZN+TF6N)*PNM                      EVAL       40
      CE(3,N,JCFCE)=UNM*E1-VNM*VOR*DNM+TF4N*XRP                              EVAL       41
      CE(4,N,JCFCE)=VNM*E1+(VNM*DNM*UNM+PNM                                  EVAL       42
     1             *(TF4N*XPHIN+TG5J*YPHIJ+TF7N))/RNM                        EVAL       43
    5 IF (L .EQ. 1) GO TO 10                                                 EVAL       44
```

```
C   ***   CHECK OF CFL CONDITION   ***                                    EVAL      45
      IF (PNM .LE. 0.) GO TO 10                                           EVAL      46
      ASQNM=ASQ(N,M)                                                      EVAL      47
      ETA=WNM*WNM-ASQNM                                                   EVAL      48
      DUM1=(ETA*(XRN*XRN+XPOR**2)+AD**2)*ASQNM                            EVAL      49
      YPRB=YPOR*BJ    $  JCF=1                                            EVAL      50
      DUM2=( (ETA+VNM*VNM)*YPRB**2)*ASQNM                                 EVAL      51
      WWX1=WNM*AA-ASQNM*XZN                                               EVAL      52
      WWX2=(WNM*BR-ASQNM*YZJ)*BJ                                          EVAL      53
      IF (DUM1 .GE. 0.) GO TO 1001                                        DUMP       2
      CALL DMPSQRT(4HEVAL,1,Z,K,M,N,DUM1)                                 DUMP       3
 1001 CONTINUE                                                            DUMP       4
      SIG=ABS(WWX1)+SQRT(DUM1)                                            EVAL      54
      SIG2=ABS(WWX2)+SQRT(DUM2)                                           EVAL      55
      IF(SIG2.LT.SIG) GO TO 7                                             EVAL      56
      JCF=2    $  SIG=SIG2                                                EVAL      57
    7 SIG3=ABS(WWX1+WWX2)                                                 EVAL      58
     1    +SQRT(DUM1+DUM2+2.*ASQNM*YPRB*(ETA*XPOR+AD*VNM))                 EVAL      59
      IF(SIG3.LT.SIG) GO TO 8                                             EVAL      60
      JCF=3    $  SIG=SIG3                                                EVAL      61
    8 CFLX=SIG/ETA                                                        EVAL      62
      IF(CFLX.LE.CFL) GO TO 10                                            EVAL      63
      NCFL=N  $ MCFL=M   $  JCFL=JCF  $  CFL=CFLX                         EVAL      64
   10 CONTINUE                                                            EVAL      65
      RETURN                                                              EVAL      66
      ENTRY EVALPR                                                        EVAL      67
C                                                                         EVAL      68
C      EVALPR - EVALSY COMPUTES CG VECTOR, WALL VALUES OF P, V2,          EVAL      69
C      V/W, S AND SHOCK SLOPES CZ AND CPHI AT PLANES                      EVAL      70
C             Y=-DY AND Y=1+DY      (EVALSY)                              EVAL      71
C             Y=-DY AND Y=1         (EVALPR)                              EVAL      72
C                                                                         EVAL      73
      YZJ=YZ(JSG) $ YPHIJ=YPHI(JSG)                                       EVAL      74
      CZY(JCG)=CZ(M)                                                      EVAL      75
      PWY(JCG)=P(1,M) $ VOWY(JCG)=V(1,M)/W(1,M) $ SWY(JCG)=SW(M)          EVAL      76
      V2(JCG)=V(1,M)+U(1,M)*BPHI(M)/B(M)                                  EVAL      77
      CPHIY(JCG)=CPHI(M)                                                  EVAL      78
      FAC=-1.                                                             EVAL      79
      GO TO 15                                                            EVAL      80
      ENTRY EVALSY                                                        EVAL      81
      FAC1=DY*TGS(IT) $ FAC=(2.-FAC1)/(2.+FAC1)                           EVAL      82
      IF (M .NE. 2) FAC=1./FAC                                            EVAL      83
      YZJ=YZ(JSG) $ YPHIJ=YPHI(JSG)                                       EVAL      84
      PWY(JCG)=P(1,M) $ VOWY(JCG)=-V(1,M)*FAC/W(1,M) $ SWY(JCG)=SW(M)     EVAL      85
      V2(JCG)=-(V(1,M)+U(1,M)*BPHI(M)/B(M))*FAC                           EVAL      86
      CZY(JCG)=CZ(M) $ CPHIY(JCG)=-CPHI(M)*FAC                            EVAL      87
   15 DO 20 N=1,NC                                                        EVAL      88
      UNM=U(N,M) $ PNM=P(N,M) $ RNM=R(N,M) $ WNM=W(N,M)                   EVAL      89
      VNM=-V(N,M)*FAC $ BB=YZJ*WNM+YPHIJ*VNM/RNM                          EVAL      90
      CG(1,N,JCG)=RB=D(N,M)*BB                                            EVAL      91
      CG(2,N,JCG)=YZJ*PNM+RB*WNM                                          EVAL      92
      CG(3,N,JCG)=RB*UNM                                                  EVAL      93
      CG(4,N,JCG)=YPHIJ*PNM/RNM+RB*VNM                                    EVAL      94
   20 CONTINUE                                                            EVAL      95
      RETURN                                                              EVAL      96
      END                                                                 EVAL      97
```

```
      SUBROUTINE JUMP(DBP,DBZ,MB)                                          JUMP      2
C                                                                          JUMP      3
C      JUMP COMPUTES JUMPS CORRESPONDING TO DISCONTINUITIES IN             JUMP      4
C      BZ AND/OR BPHI.                                                     JUMP      5
C      NOTE THAT THE KEY IJUMP1(MB), USED IN SUBROUTINE WALL,              JUMP      6
C      IS SET IN THIS SUBROUTINE                                           JUMP      7
C          IJUMP1(MB) = 2 MEANS THERE IS NO JUMP IN PW                     JUMP      8
C                     = 3 MEANS THERE IS AN EXPANSION JUMP                 JUMP      9
C                     = 4 MEANS THERE IS A COMPRESSION JUMP                JUMP     10
C                                                                          JUMP     11
      COMMON NC,MC,K,PINF,DINF,PHI0,IDYAW,PI,RAD                           NEWCOM    1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                  NEWCOM    2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)       NEWCOM    3
      COMMON CU(4,20,25),CUP(4,20,25)                                      NEWCOM    4
C     ***   END OF BLANK COMMON   ***                                      CD3CSS   32
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                         CBODY     2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)              CBODY     3
      COMMON /BLK01/ CZY(3),CPHIY(3),V2(3),VOWY(3),PWY(3),SWY(3)           BLK01     2
     1   ,UNOR(3,2),CF(4,20,2),CG(4,20,3),CE(4,20,2)                       BLK01     3
      COMMON /BLK02/ THETA,DY,TG4(3),TG5(3),TG6(3)                         BLK02     2
     1   ,X(20),XZ(20,2),XR(20,2),XPHI(20,2),Y(25)                         BLK02     3
     2   ,TF4(20,2),TF6(20,2),TF7(20,2)                                    BLK02     4
      COMMON /BLK03/ ICFL,A2(3),A3(3),A4(3),A5(3),A7(3)                    BLK03     2
     1   ,IJUMP(25),IJUMP1(25),IJMPKT(25)                                  BLK03     3
      COMMON /BLK04/ GAMMA,GB,GD,GE,GA2,DDX,DDY,HOT2,ELIM,LCNT,ISWSMO,NA   BLK04     2
     1   ,SW(25),GM(20,25)                                                 BLK04     3
      COMMON/RGASS/AX,HX,TX,RRX,GX,NTEST,NGAS,NFIRST                       JUMP     13
C                                                                          JUMP     14
      ICNT=0                                                               JUMP     15
      ERR2=SIN2=0.0                                                        JUMP     16
      DWC=.05                                                              JUMP     17
      DATA (INT=18)                                                        JUMP     18
      VW=V(1,MB) $ PW=P(1,MB) $ DW=D(1,MB) $ WW=W(1,MB)                    JUMP     19
      ASQW=ASQ(1,MB) $ UW=U(1,MB)                                          JUMP     20
      SO=SW(MB)                                                            JUMP     21
      PHID=PHI(MB)/RAD                                                     JUMP     22
      WRITE (6,3100) MB,PHID,K,Z                                           JUMP     23
 3100 FORMAT(1H0,*JUMP IS CALLED FOR PLANE*,I4,5X,*PHI IS*,F7.2,           JUMP     24
     1   5X,*K IS*,I4,5X,*Z IS*,1PE15.6)                                   JUMP     25
      WRITE (6,3110)                                                       JUMP     26
 3110 FORMAT(1H ,30X,*THE INPUT VARIABLES ARE AS FOLLOWS*)                 JUMP     27
      WRITE (6,3120) PW,DW,UW,VW,WW,SO,ASQW                                JUMP     28
 3120 FORMAT(1H ,22HP,D,U,V,W,S,ASQ          ,1P7E15.5)                   JUMP     29
      WRITE (6,3130) B(MB),BZ(MB),BPHI(MB),DBP,DBZ,HOT2                    JUMP     30
 3130 FORMAT(1H ,22HB,BZ,BPHI,DBP,DBZ,HOT2,1P6E15.5)                       JUMP     31
      ETAP=BZ(MB) $ SIP=BPHI(MB)/B(MB)                                     JUMP     32
      ETAM=ETAP+DBZ $ SIM=SIP+DBP                                          JUMP     33
      XNP2=1.+ETAP**2+SIP**2 $ XNP=SQRT(XNP2)                              JUMP     34
      XNM2=1.+ETAM**2+SIM**2 $ XNM=SQRT(XNM2)                              JUMP     35
      XNMP=1.+ETAP*ETAM+SIP*SIM                                            JUMP     36
      DUM2=XNP2*XNM2-XNMP**2 $ DUM=SQRT(DUM2) $ XT2=DBP*ETAM-SIM*DBZ       JUMP     37
      XTSQ=DBP**2+DBZ**2+XT2**2 $ XT=SQRT(XTSQ)                            JUMP     38
      QT=(XT2*UW+DBP*WW-DBZ*VW)/XT                                         JUMP     39
      QSM=XNM*(UW-ETAP*WW-SIP*VW)/DUM $ AMMSQ=QSM**2/ASQW                  JUMP     40
      TEST=(XNMP*ETAM-XNM2*ETAP)*QSM                                       JUMP     41
      IF (TEST .GT. 0.) GO TO 5                                            JUMP     42
C                                                                          JUMP     43
```

```
C       THE FLOW DOES NOT CROSS EDGE FROM MINUS TO PLUS                    JUMP       44
C           (NO JUMPS IN P, D, S, QSQ)                                     JUMP       45
C                                                                          JUMP       46
      WRITE (6,3135)                                                       JUMP       47
 3135 FORMAT(1H ,*THE FLOW DOES NOT CROSS EDGE IN MARCHING DIRECTION*)     JUMP       48
      IJUMP1(MB)=2                                                         JUMP       49
      QSP=QSM                                                              JUMP       50
      GO TO 110                                                            JUMP       51
    5 THETAR=ACOS(XNMP/(XNP*XNM)) $ CONST=HOT2-QT**2 $ QSQ=QSM**2          JUMP       52
      THETAD=THETAR/RAD $ AMACH=SQRT(QSQ/ASQW)                             JUMP       53
      IF (AMMSQ .GE. 1.0) GO TO 10                                         JUMP       54
C   ***   SUBSONIC CORNER FLOW (NO JUMPS IN P,D,S,QSQ)   ***               JUMP       55
      IJUMP1(MB)=2                                                         JUMP       56
      QSP=QSM $ GO TO 110                                                  JUMP       57
   10 IF (QSM .LT. 0.) GO TO 20                                            JUMP       58
C   ***   SUPERSONIC EXPANSION CORNER   ***                              JUMP       59
      WRITE (6,3140)                                                       JUMP       60
 3140 FORMAT(1H ,20X,*SUPERSONIC EXPANSION CORNER WHERE*)                  JUMP       61
      WRITE (6,3150) THETAD,AMACH,QT,QSM                                   JUMP       62
 3150 FORMAT(1H ,22HTHETA,AMACH,QT,QSM    ,1P4E15.5)                      JUMP       63
      CALL RGAS(PW,DW,DUMMY,4)                                             JUMP       64
      VPOVR=SQRT(ASQW/(CONST-2.*HX-ASQW))                                  JUMP       65
      DEL=THETAR/FLOAT(INT)                                                JUMP       66
      DO 15 I=1,INT                                                        JUMP       67
      CC=0. $ PWO=PW                                                       JUMP       68
      DWO=DW                                                               JUMP       69
   17 DPDAL=-DW*QSQ*VPOVR                                                  JUMP       70
      DDDAL=DPDAL/ASQW                                                     JUMP       71
      CD=CC+1.                                                             JUMP       72
      PW=(PWO+PW*CC+DEL*DPDAL)/CD                                          JUMP       73
      DW=(DWO+DW*CC+DEL*DDDAL)/CD                                          JUMP       74
      CALL RGAS(PW,DW,DUMMY,4)                                             JUMP       75
      QSQ=CONST-2.*HX                                                      JUMP       76
      ASQW=AX*AX                                                           JUMP       77
      VPOVR=1./SQRT(QSQ/ASQW-1.0)                                          JUMP       78
      CC=CD $ IF (CC .LT. 1.5) GO TO 17                                    JUMP       79
   15 CONTINUE                                                             JUMP       80
      ICFL=1 $ IJUMP1(MB)=3 $ IJMPKT(MB)=1                                 JUMP       81
      QSP=SQRT(QSQ) $ GO TO 100                                            JUMP       82
C   ***   SUPERSONIC COMPRESSION CORNER          ***                      JUMP       83
   20 COSTH2=(XNMP/(XNP*XNM))**2                                           JUMP       84
      IJUMP1(MB)=4 $ IJMPKT(MB)=1                                          JUMP       85
      ISWSMO=0                                                             JUMP       86
      WRITE (6,3160)                                                       JUMP       87
 3160 FORMAT(1H ,20X,*SUPERSONIC COMPRESSION CORNER WHERE*)                JUMP       88
      WRITE (6,3150) THETAD,AMACH,QT,QSM                                   JUMP       89
      IF (NTEST .LT. 0) GO TO 4000                                         JUMP       90
C   ***   (PERFECT GAS OBLIQUE SHOCK RELATIONS)   ***                      JUMP       91
      SINTH2=1.-COSTH2                                                     JUMP       92
      AM4=AMMSQ**2 $ AM2=AMMSQ                                             JUMP       93
      C1=-((AM2+2.)/AM2+GAMMA*SINTH2) $ C3=-COSTH2/AM4                     JUMP       94
      C2=(2.*AM2+1.)/AM4+(.25*(GAMMA+1.)**2+(GAMMA-1.)/AM2)*SINTH2         JUMP       95
      DUMM=C1/3. $ A=-C2+DUMM*C1 $ SB=C3-(C2-2.*C1**2/9.)*DUMM             JUMP       96
      DDUM=SQRT(A/3.) $ DDUM1=2.*DDUM                                      JUMP       97
      TEST=-.5*SB/(DDUM**3) $ IF (TEST .GE. -1.0) GO TO 25                 JUMP       98
      WRITE (6,3165)                                                       JUMP       99
 3165 FORMAT(1H ,20X,*NORMAL SHOCK MODE IS USED*)                          JUMP      100
```

```
      PWO=PW $ T=AM2 $ GO TO 45                                          JUMP      101
   25 XX=ACOS(TEST)/3.                                                   JUMP      102
      X1=COS(XX) $ X2=COS(XX+2.*PI/3.) $ X3=COS(XX+4.*PI/3.)             JUMP      103
      IF (X1 .LT. X2) GO TO 30 $ XDUM=X1 $ X1=X2 $ X2=XDUM               JUMP      104
   30 IF (X1 .LE. X3) GO TO 35 $ SX=X1 $ GO TO 40                        JUMP      105
   35 IF (X3 .LE. X2) X2=X3 $ SX=X2                                      JUMP      106
   40 SINTH2=DDUM1*SX-DUMM $ T=AM2*SINTH2                                JUMP      107
      G9=GE/GD                                                           JUMP      108
   45 PW=PW*((GA2*T-1.)/G9)                                              JUMP      109
      DW=DW*(T/(T/G9+1./GE))                                             JUMP      110
      QSP=QSM*SQRT(1.-(T-1.0)*(GAMMA*T+1.)/(T*AM2*GE**2))                JUMP      111
      CALL RGAS(PW,DW,SW(MB),4)                                          JUMP      112
      ASQW=AX*AX                                                         JUMP      113
      IF (TEST .GE. -1.0) GO TO 100                                      JUMP      114
      CONST=ASQW/(GAMMA-1.0)+.5*QSP**2                                   JUMP      115
      PW=PW*SINTH2                                                       JUMP      116
      IF (PW .GT. PWO) GO TO 75                                          JUMP      117
      PW=PWO $ IJUMP1(MB)=2 $ IJMPKT(MB)=0                               JUMP      118
      GO TO 75                                                           JUMP      119
 4000 CONTINUE                                                           JUMP      120
C    ***   REAL GAS OBLIQUE SHOCK    ***                                JUMP      121
      TEST=0.0                                                           JUMP      122
      CALL RGAS(PW,DW,DUMMY,4)                                           JUMP      123
      WRITE(6,300)PW,DW,AX,HX,ASQW,AMMSQ                                 JUMP      124
  300 FORMAT(1H ,6E16.8)                                                 JUMP      125
      HXM=HX                                                             JUMP      126
      SIND=SQRT(1.-COSTH2)                                               JUMP      127
      TAND=SIND/SQRT(COSTH2)                                             JUMP      128
      SIN1=-DWC+.001+SIN(1./SQRT(AMMSQ))                                 JUMP      129
      IF(SIND.GE.SIN1)SIN1=SIND+.001-DWC                                 JUMP      130
  111 SIN1=SIN1+DWC                                                      JUMP      131
      GO TO 108                                                          JUMP      132
  101 SIN1=SIN2-ERR2*(SIN3-SIN2)/(ERR3-ERR2)                             JUMP      133
  108 COS1=SQRT(1.-SIN1*SIN1)                                            JUMP      134
      CTN1=COS1/SIN1                                                     JUMP      135
      U1=QSM*SIN1                                                        JUMP      136
      VT=QSM*COS1                                                        JUMP      137
      U2=VT*(1.-TAND*CTN1)/(TAND+CTN1)                                   JUMP      138
      DWT=DW*U1/U2                                                       JUMP      139
      PWT=PW+DW*U1*U1-DWT*U2*U2                                          JUMP      140
      WRITE(6,300)PWT,DWT,SIN1                                           JUMP      141
      CALL RGAS(PWT,DWT,SW(MB),4)                                        JUMP      142
      QSP=-SQRT(U2*U2+VT*VT)                                             JUMP      143
      IF(-QSP/AX.GE.1.)GO TO 109                                         JUMP      144
      IF(DWC.LT..005)GO TO 113                                           JUMP      145
      DWC=DWC/2.                                                         JUMP      146
      SIN1=SIN1-DWC                                                      JUMP      147
      GO TO 108                                                          JUMP      148
  113 WRITE(6,3000)                                                      JUMP      149
 3000 FORMAT(* NO OBLIQUE SHOCK SOLUTION. NORMAL SHOCK MODE USED.*)      JUMP      150
      GAMMA=GM(1,MB)                                                     JUMP      151
      GA2=2.*GAMMA/(GAMMA-1.)                                            JUMP      152
      GCC=(GAMMA+1.)/(GAMMA-1.)                                          JUMP      153
      GE=(GAMMA+1.)/2.                                                   JUMP      154
      TEST=-2.                                                           JUMP      155
      PWT=PW*((GA2*AMMSQ-1.)/GCC)                                        JUMP      156
      DWT=DW*(AMMSQ/(AMMSQ/GCC+1./GE))                                   JUMP      157
```

```
      CALL RGAS(PWT,DWT,SW(MB),4)                                        JUMP  158
      ASQW=AX*AX                                                         JUMP  159
      GO TO 98                                                           JUMP  160
  109 ERR1=2.*(HXM-HX)*U1*U1-U2*U2                                       JUMP  161
      ASQW=AX*AX                                                         JUMP  162
      WRITE(6,2000)ICNT,SIN1,ERR1,SIN2,ERR2,SIN3,ERR3                    JUMP  163
 2000 FORMAT(1H ,I5,6E16.8)                                              JUMP  164
      IF(ABS(ERR1/HX).LT.ELIM)GO TO 98                                   JUMP  165
      ICNT=ICNT+1                                                        JUMP  166
      IF(ICNT.GT.2)GO TO 106                                             JUMP  167
      ERR3=ERR2                                                          JUMP  168
      SIN3=SIN2                                                          JUMP  169
      SIN2=SIN1                                                          JUMP  170
      ERR2=ERR1                                                          JUMP  171
      GO TO 111                                                          JUMP  172
  106 IF(ICNT.LE.2*LCNT)GO TO 103                                        JUMP  173
      WRITE(6,200)                                                       JUMP  174
  200 FORMAT(1H1,*OBLIQUE SHOCK ITERATION EXCEEDS LIMIT*)                JUMP  175
      STOP                                                               JUMP  176
  103 IF(ERR1*ERR2.GE.0.0)GO TO 104                                      JUMP  177
      ERR3=ERR1                                                          JUMP  178
      SIN3=SIN1                                                          JUMP  179
      GO TO 101                                                          JUMP  180
  104 IF(ERR1*ERR3.GE.0.0)GO TO 105                                      JUMP  181
      ERR2=ERR1                                                          JUMP  182
      SIN2=SIN1                                                          JUMP  183
      GO TO 101                                                          JUMP  184
  105 ERR3=ERR2                                                          JUMP  185
      ERR2=ERR1                                                          JUMP  186
      SIN3=SIN2                                                          JUMP  187
      SIN2=SIN1                                                          JUMP  188
      GO TO 111                                                          JUMP  189
   98 CONTINUE                                                           JUMP  190
      DW=DWT                                                             JUMP  191
      PW=PWT                                                             JUMP  192
      IF (TEST .GE. -1.0) GO TO 100                                      JUMP  193
      CONST=HX+.5*QSP**2                                                 JUMP  194
      PW=PW *(1.-COSTH2)                                                 JUMP  195
      IF (PW .GT. PW0) GO TO 75                                          JUMP  196
      PW=PW0 $ IJUMP1(MB)=2 $ IJMPKT(MB)=0                               JUMP  197
   75 CALL RGAS(PW,DW,SW(MB),5)                                          JUMP  198
      ASQW=AX*AX                                                         JUMP  199
      QSP=-SQRT(2.*(CONST-HX))                                           JUMP  200
  100 D(1,MB)=DW $ P(1,MB)=PW $ ASQ(1,MB)=ASQW                           JUMP  201
  110 AA1=QT/XT $ AA2=QSP/(DUM*XNP)                                      JUMP  202
      V(1,MB)=AA2*(XNP2*SIM-XNMP*SIP)-AA1*DBZ                            JUMP  203
      U(1,MB)=AA1*XT2+AA2*(XNMP-XNP2)                                    JUMP  204
      W(1,MB)=AA1*DBP+AA2*(XNP2*ETAM-XNMP*ETAP)                          JUMP  205
      WRITE (6,3170)                                                     JUMP  206
 3170 FORMAT(1H ,30X,*THE OUTPUT VARIABLES ARE AS FOLLOWS*)              JUMP  207
      WRITE (6,3120) PW,DW,U(1,MB),V(1,MB),W(1,MB),SW(MB),ASQ(1,MB)      JUMP  208
      RETURN                                                             JUMP  209
      END                                                                JUMP  210
```

```
      SUBROUTINE TRANF(M,J,I)                                                  TRANF     2
C                                                                              TRANF     3
C      TRANF DEFINES QUANTITIES ASSOCIATED WITH THE CLUSTERING                 TRANF     4
C      TRANSFORMATION IN THE R DIRECTION (SEE STATEMENTS                       TRANF     5
C      1-8 BELOW).  THE CLUSTERING TRANSFORMATION IS ASSUMED                   TRANF     6
C      IN THE FORM                                                             TRANF     7
C            SX=SF(X,Y,Z)   WHERE   SX=(R-B(Z,PHI))/(C(Z,PHI)-B(Z,PHI))       TRANF     8
C      THE USER CAN SPECIFY THE FUNCTION SF(X,Y,Z).                            TRANF     9
C      THE USER MUST DEFINE AS FUNCTIONS OF (X,Y,Z) THE FOLLOWING              TRANF    10
C            SX, SFX, SFY, SFZ, SFXX, SFZX, SFYX                               TRANF    11
C      SEE USERS MANUAL FOR RESTRICTIONS AND INSTRUCTIONS                      TRANF    12
C      J=1,2,3 IS A LINE INDEX FOR TRANF QUANTITIES                            TRANF    13
C      I=1,2   IS A LINE INDEX FOR TRANF QUANTITIES                            TRANF    14
C                                                                              TRANF    15
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                               NEWCOM    1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                      NEWCOM    2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)           NEWCOM    3
      COMMON CU(4,20,25),CUP(4,20,25)                                          NEWCOM    4
C   ***   END OF BLANK COMMON   ***                                            CD3CSS   32
      COMMON /CTRANF/ NSFD,SFD(20),SFXD(20),SFXXD(20)                          NEWCOM    8
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                             CBODY     2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)                  CBODY     3
      COMMON /BLK02/ THETA,DY,TG4(3),TG5(3),TG6(3)                             BLK02     2
     1   ,X(20),XZ(20,2),XR(20,2),XPHI(20,2),Y(25)                             BLK02     3
     2   ,TF4(20,2),TF6(20,2),TF7(20,2)                                        BLK02     4
C                                                                              TRANF    18
      CZM=CZ(M)     $  BM=B(M)     $   BZM=BZ(M)      $BPHIM=BPHI(M)           TRANF    19
      CMB=C(M)-BM     $   BZMCZ=BZM - CZM  $  BPMCP=BPHIM  -CPHI(M)           TRANF    20
      YZJ=YZ(J)     $  YPHIJ=YPHI(J)    $  TG6J=TG6(J)                       TRANF    21
C*************************************************************************** TRANF    22
C                                                                              TRANF    23
C      THIS ROUTINE IS VERSION 1 OF TRANF CORRESPONDING EITHER                 TRANF    24
C      TO NO CLUSTERING, I.E., SF(X,Y,Z)=X                                     TRANF    25
C      OR THE USER HAS READ IN THE SF(X) DATA POINTS                           TRANF    26
C                                                                              TRANF    27
      SFX=1.0     $  SFXX=0.0     $  SFY=0.0                                   TRANF    28
      SFZX=0.0     $  SFYX=0.0     $  SFZ=0.0                                  TRANF    29
      DO 100 N=1,NC                                                            TRANF    30
      IF (NSFD .EQ. 0) GO TO 25                                                TRANF    31
C                                                                              TRANF    32
C      THE USER READ IN THE SF(X) DATA POINTS                                  TRANF    33
C                                                                              TRANF    34
      SX=SFD(N) $ SFX=SFXD(N) $ SFXX=SFXXD(N)                                  TRANF    35
      GO TO 50                                                                 TRANF    36
C                                                                              TRANF    37
C      CORRESPONDS TO NO CLUSTERING                                            TRANF    38
C                                                                              TRANF    39
   25 SX=X(N)                                                                  TRANF    40
C   ***   THE FOLLOWING STATEMENTS SHOULD APPEAR IN ALL VERSIONS   ***         TRANF    41
   50 SX1=SX-1.                                                                TRANF    42
      FX=1./SFX                                                                TRANF    43
      FTHD=-SFY*FX*YPHIJ                                                       TRANF    44
      FZ=-FX*(SFZ+SFY*YZJ)                                                     TRANF    45
   1  R(N,M)=BM+SX*CMB                                                         TRANF    46
   2  XR(N,I)=TXR=FX/CMB                                                       TRANF    47
   3  XZ(N,I)=FZ+TXR*(SX1*BZM-SX*CZM)                                          TRANF    48
   4  XPHI(N,I)=FTHD+TXR*(SX1*BPHIM-SX*CPHI(M))                                TRANF    49
```

```
  5    TF4(N,I)=SFXX/SFX                                                  TRANF     50
  7    TF6(N,I)=TG6J +(SFZX+SFYX*YZJ)/SFX-BZMCZ/CMB                       TRANF     51
  8    TF7(N,I)=SFYX*YPHIJ/SFX-BPMCP/CMB                                  TRANF     52
100    CONTINUE                                                           TRANF     53
       RETURN                                                             TRANF     54
       ENTRY TRANFW                                                       TRANF     55
       IVERSON=1                                                          TRANF     56
       WRITE (6,3000) IVERSON                                             TRANF     57
       IF (NSFD .EQ. 0) WRITE (6,3010)                                    TRANF     58
       IF (NSFD .NE. 0) WRITE (6,3020)                                    TRANF     59
3000   FORMAT(1H0,20X,*PROGRAM TRANF*,6X,*VERSION*,I4)                    TRANF     60
3010   FORMAT(11X,*EQUAL SPACING IN RADIAL DIRECTION*)                    TRANF     61
3020   FORMAT(11X,*SF(X) WAS READ IN AS DATA POINTS*)                     TRANF     62
       RETURN                                                             TRANF     63
       END                                                                TRANF     64
```

```
      SUBROUTINE TRANG(YY,M,J)                                          TRANG     2
C                                                                       TRANG     3
C      TRANG DEFINES QUANTITIES ASSOCIATED WITH THE CLUSTERING          TRANG     4
C      TRANSFORMATION IN THE PHI DIRECTION (SEE STATEMENTS__            TRANG     5
C      1-6 BELOW).  THE CLUSTERING TRANSFORMATION IS ASSUMED            TRANG     6
C      IN THE FORM                                                      TRANG     7
C             THETA=SG(YY,Z)    WHERE    THETA=PHI/PHIO                TRANG     8
C      THE USER CAN SPECIFY THE FUNCTION SG(YY,Z)                       TRANG     9
C      THE USER MUST DEFINE AS FUNCTIONS OF (YY,Z) THE FOLLOWING        TRANG    10
C             SG, SGY, SGZ, SGYY, SGYZ                                  TRANG    11
C      SEE USERS MANUAL FOR RESTRICTIONS AND INSTRUCTIONS               TRANG    12
C      M IS THE INDEX FOR THE TANGENTIAL PLANE                          TRANG    13
C      J=1,2,3 IS A LINE INDEX FOR TRANG QUANTITIES                     TRANG    14
C                                                                       TRANG    15
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                        NEWCOM    1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)               NEWCOM    2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)    NEWCOM    3
      COMMON CU(4,20,25),CUP(4,20,25)                                   NEWCOM    4
C   ***   END OF BLANK COMMON   ***                                     CD3CSS   32
      COMMON /CTRANG/ NSGD,SGD(25),SGYD(25),SGYYD(25)                   NEWCOM    6
     1   ,GYMDY,GYYMDY,GY1PDY,GYY1PDY                                   CTRANG    3
     2   ,MCP                                                           NEWCOM    7
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                      CBODY     2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)           CBODY     3
      COMMON /BLK02/ THETA,DY,TG4(3),TG5(3),TG6(3)                      BLK02     2
     1   ,X(20),XZ(20,2),XR(20,2),XPHI(20,2),Y(25)                      BLK02     3
     2   ,TF4(20,2),TF6(20,2),TF7(20,2)                                 BLK02     4
C                                                                       TRANG    18
C      THIS ROUTINE IS VERSION 1 OF TRANG CORRESPONDING EITHER          TRANG    19
C      TO NO CLUSTERING, I.E., SG(YY,Z)=YY                              TRANG    20
C      OR THE USER HAS READ IN THE PHI"S                                TRANG    21
C                                                                       TRANG    22
      IF (NSGD .EQ. 0) GO TO 50                                         TRANG    23
C                                                                       TRANG    24
C      THE USER READ IN THE PHI"S                                       TRANG    25
C                                                                       TRANG    26
      IF (M .NE. MCP) GO TO 30                                          TRANG    27
      SGY=GY1PDY $ SGYY=GYY1PDY $ SGZ=SGYZ=0.                           TRANG    28
      GO TO 2                                                           TRANG    29
   30 IF (M .NE. 0) GO TO 35                                            TRANG    30
      SGY=GYMDY $ SGYY=GYYMDY $ SGZ=SGYZ=0.                             TRANG    31
      GO TO 2                                                           TRANG    32
   35 SG=SGD(M) $ SGY=SGYD(M) $ SGYY=SGYYD(M) $ SGZ=SGYZ=0.             TRANG    33
      GO TO 1                                                           TRANG    34
C                                                                       TRANG    35
C      CORRESPONDS TO NO CLUSTERING                                     TRANG    36
C                                                                       TRANG    37
   50 SG=YY $ SGY=1. $ SGZ=SGYY=SGYZ=0.                                 TRANG    38
C                                                                       TRANG    39
C   ***   THE FOLLOWING STATEMENTS SHOULD APPEAR IN ALL VERSIONS   ***  TRANG    40
    1  THETA=SG                                                        TRANG    41
    2  YPHI(J)=1.0/(PHIO*SGY)                                          TRANG    42
    3  YZ(J)=-SGZ/SGY                                                  TRANG    43
    4  TG4(J)=SGY                                                      TRANG    44
    5  TG5(J)=SGYY/SGY                                                 TRANG    45
    6  TG6(J)=SGYZ/SGY                                                 TRANG    46
      RETURN                                                            TRANG    47
```

```
      ENTRY TRANGW                                                      TRANG     48
      IVERSON=1                                                         TRANG     49
      WRITE (6,3000) IVERSON                                            TRANG     50
      IF (NSGD .EQ. 0) WRITE (6,3010)                                   TRANG     51
      IF (NSGD .NE. 0) WRITE (6,3020)                                   TRANG     52
 3000 FORMAT(1H0,20X,*PROGRAM TRANG*,6X,*VERSION*,I4)                   TRANG     53
 3010 FORMAT(11X,*EQUAL SPACING IN TANGENTIAL DIRECTION*)               TRANG     54
 3020 FORMAT(11X,*THE PHI"S WERE READ IN BY THE USER*)                  TRANG     55
      RETURN                                                            TRANG     56
      END                                                               TRANG     57
```

```
      SUBROUTINE WALL(M,JR,JL,JSG,IF,L)                                  WALL      2
C                                                                        WALL      3
C      WALL COMPUTES PREDICTED OR CORRECTED Z DERIVATIVES OF             WALL      4
C      P, V2, AND S(ENTROPY) USING CHARACTERISTIC COMP. RELS.            WALL      5
C      V2 IS VEL. COMP. TANGENT TO WALL      V2=V+(BPHI/B)*U             WALL      6
C      V2(J), SW(J), VOWY(J), AND PWY(J) ARE WALL VALUES OF              WALL      7
C      V2, S, V/W, AND P RESP. WHERE J=JR OR JL                          WALL      8
C      JR AND JL ARE LINE IDENT. INDEXES FOR Y DIFFS.                    WALL      9
C      JSG=1,2,3 LINE INDEX FOR TRANS AND BODY PARAMETERS                WALL     10
C      IF=1,2 LINE INDEX FOR TRANF PARAMETERS                            WALL     11
C      L = 0 CORRESPONDS TO PREDICTOR                                    WALL     12
C        = 1 CORRESPONDS TO CORRECTOR                                    WALL     13
C      THIS VERSION OF WALL CONTAINS SEVERAL OPTIONS FOR WALL B.C.       WALL     14
C          ISWSMO NE 0 MEANS WALL ENTROPY EXTRAPOLATION                  WALL     15
C          MOD1 = 1 MEANS SECOND ORDER ACCURACY                          WALL     16
C          ISWMOD = 0 MEANS MOD 0 FOR WALL B.C.                          WALL     17
C                 = 3 MEANS MOD 3 FOR WALL B.C.                          WALL     18
C      THIS ROUTINE CONTAINS SPECIAL FEATURES AFTER A JUMP               WALL     19
C          IJUMP1(M) = 0 MEANS NO JUMP ON LINE                           WALL     20
C          IJUMP1(M) NE 0 MEANS JUMP HAS BEEN CALLED (SEE JUMP)          WALL     21
C          IJUMP1(M) = 2 MEANS NO SECOND ORDER ACCURACY                  WALL     22
C                        AND NO ENTROPY EXTRAP. IF A COMPRESSION JUMP    WALL     23
C                        AND MOD 0 FOR WALL B.C.                         WALL     24
C                                                                        WALL     25
      COMMON/RGASS/AX,HX,TX,RRX,GX,NTEST,NGAS,NFIRST                     WALL     26
      COMMON NC,MC,K,PINF,DINF,PHI0,IDYAW,PI,RAD                         NEWCOM    1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                NEWCOM    2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)     NEWCOM    3
      COMMON CU(4,20,25),CUP(4,20,25)                                    NEWCOM    4
C     ***   END OF BLANK COMMON   ***                                    CD3CSS   32
      COMMON /CWALL/ PZ,V2Z,SZ,ISWMOD,MOD1,NJMPKT,NJMKTC,KCFL,KFAC       CWALL     2
     1   ,PZCOR(25)                                                      CWALL     3
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                       CBODY     2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)            CBODY     3
      COMMON /BLK01/ CZY(3),CPHIY(3),V2(3),VOWY(3),PWY(3),SWY(3)         BLK01     2
     1   ,UNOR(3,2),CF(4,20,2),CG(4,20,3),CE(4,20,2)                     BLK01     3
      COMMON /BLK02/ THETA,DY,TG4(3),TG5(3),TG6(3)                       BLK02     2
     1   ,X(20),XZ(20,2),XR(20,2),XPHI(20,2),Y(25)                       BLK02     3
     2   ,TF4(20,2),TF6(20,2),TF7(20,2)                                  BLK02     4
      COMMON /BLK03/ ICFL,A2(3),A3(3),A4(3),A5(3),A7(3)                  BLK03     2
     1   ,IJUMP(25),IJUMP1(25),IJMPKT(25)                                BLK03     3
      COMMON /BLK04/ GAMMA,GR,GD,GE,GA2,DDX,DDY,HOT2,ELIM,LCNT,ISWSMO,NA BLK04     2
     1   ,SW(25),GM(20,25)                                               BLK04     3
C                                                                        WALL     28
      DIMENSION DCGY(4)                                                  WALL     29
      BET=BETA                                                           WALL     30
      KMOD1=MOD1 $ KSWMOD=ISWMOD                                         WALL     31
      BM=B(M) $ BZM=BZ(M) $ BPHOB=BPHI(M)/BM                             WALL     32
      YPHIJ=YPHI(JSG) $ YZJ=YZ(JSG) $ XRW=XR(1,IF)                       WALL     33
      UW=U(1,M) $ WW=W(1,M) $ PW=P(1,M)                                  WALL     34
      VW=V(1,M) $ DW=D(1,M) $ ASQW=ASQ(1,M)                              WALL     35
      VOR=VW/BM $ VOW=VW/WW $ YPOR=YPHIJ/BM                              WALL     36
      BB=WW*YZJ+YPHIJ*VOR                                                WALL     37
      PX=(P(2,M)-PW)*DDX                                                 WALL     38
      DWW=DW*WW $ ETA=WW**2-ASQW                                         WALL     39
      CDUMP=(ETA*(1.+BPHOB**2)+(WW*BZM)**2)/ASQW                         DUMP     33
      IF (CDUMP .GE. 0.) GO TO 1001                                      DUMP     34
```

```
      CALL DMPSQRT(4HWALL,1,Z,K,M,1,CDUMP)                                  DUMP      35
 1001 BETA=SQRT(CDUMP)                                                      DUMP      36
      ALAM=ASQW*(BETA-BZM)/ETA $ DUM4=VW+BPHOR*UW                           WALL      41
      AAX=UNOR(2,IF)*DDX $ DSY=(SWY(JR)-SWY(JL))*DDY                        WALL      42
      V2Y=(V2(JR)-V2(JL))*DDY                                               WALL      43
      IF (IJUMP1(M) .EQ. 0) GO TO 20                                        WALL      44
      IF (IJUMP1(M) .EQ. 2) GO TO 15                                        WALL      45
      IF (L .EQ. 1) GO TO 10                                                WALL      47
  200 IF (IJUMP1(M) .NE. 3) GO TO 210                                       WALL      54
      IF (IJMPKT(M) .GT. NJMPKT) GO TO 230                                  WALL      55
      IFAC=NJMPKT                                                           PXWALL     3
      GO TO 215                                                             WALL      56
  210 IF (IJMPKT(M) .GT. NJMKTC) GO TO 230                                  WALL      57
      IFAC=NJMKTC                                                           PXWALL     4
      IF (ICFL*KCFL .NE. 0) GO TO 215                                       WALL      58
      IJMPKT(M)=IJMPKT(M)+KFAC $ GO TO 10                                   WALL      59
  215 IJMPKT(M)=IJMPKT(M)+1 $ GO TO 10                                      WALL      60
  230 IJUMP1(M)=2 $ IJMPKT(M)=0 $ GO TO 15                                  WALL      61
   10 FAC=FLOAT(IJMPKT(M)-1)/FLOAT(IFAC)                                    PXWALL     5
      PX=FAC*PX $ AAX=FAC*AAX                                               PXWALL     6
   15 MOD1=0 $ ISWMOD=0                                                     WALL      63
   20 CONTINUE                                                              WALL      64
      IF (ISWMOD .EQ. 3) GO TO 25                                           WALL      65
      DVOWY=(VOWY(JR)-VOWY(JL))*DDY $ PY=(PWY(JR)-PWY(JL))*DDY              WALL      66
      DUMM=(BZM/BM+YPOR*DVOWY)*WW $ DUM=BB*WW/ASQW-YZJ                      WALL      67
      PZ23=(ALAM*DUM+BZM*YZJ+YPOR*BPHOB)*PY                                 WALL      68
     1     -DWW*(BR*(A5(JSG)+VOW*A3(JSG))                                   WALL      69
     2     -ALAM*(DUMM+(TF6(1,IF)-TG6(JSG))*WW+VOR*TF7(1,IF)))              WALL      70
      V2Z23=(BB*(V2Y-UW*A3(JSG))+YPOR*PY/DW)/WW                             WALL      71
      GO TO 50                                                              WALL      72
   25 TG5J=TG5(JSG)                                                         WALL      73
      DO 40 I=1,4                                                           WALL      74
      DCGY(I)=(CG(I,1,JR)-CG(I,1,JL))*DDY                                   WALL      75
   40 CONTINUE                                                              WALL      76
      CE1=DW*(UW/BM+TG5J*BB+TF6(1,IF)*WW+TF7(1,IF)*VOR)                     WALL      77
      DUM1=(TG5J*YZJ+TG6(JSG))*PW $ DUM2=(TG5J*YPHIJ+BPHOB)*PW/BM           WALL      78
      QSQ=WW**2+VW**2+UW**2                                                 WALL      79
      DUM3=UW*DCGY(3)-QSQ*DCGY(1)+WW*(DCGY(2)+DUM1)+VW*(DCGY(4)+DUM2)       WALL      80
C    ***    THE EQUATION FOR XK1 IS VALID FOR PERFECT GAS ONLY    ***       WALL      81
      XK1=-DW/(ASQW*GB)                                                     WALL      82
      IF(NTEST.GE.0)GO TO 60                                                WALL      83
      DX=1.025*DW                                                           WALL      84
      CALL RGAS(PW,DX,DUMMY,4)                                              WALL      85
      HX1=HX                                                                WALL      86
      DX=.975*DW                                                            WALL      87
      CALL RGAS(PW,DX,DUMMY,4)                                              WALL      88
      XK1=.05*DW/(HX1-HX)                                                   WALL      89
   60 CONTINUE                                                              WALL      90
      PZ23=ALAM*WW*(CE1+2.*DCGY(1)+XK1*DUM3/DW)                             WALL      91
     1     +(BZM-ALAM)*(DCGY(2)+DUM1)-DCGY(3)+BPHOB*(DCGY(4)+DUM2)          WALL      92
      V2Z23=(BPHOB*DCGY(3)-DUM4*DCGY(1)+DCGY(4)+DUM2)/DWW                   WALL      93
   50 PZ=ALAM*XRW*PX-(DWW*(ALAM*AAX-WW*A7(JSG)                              WALL      94
     1   -VW*A4(JSG)+DUM4*VOW/BM)+PZ23)/BETA                                WALL      95
      V2Z=UW*A4(JSG)-VOR*BZM-V2Z23                                          WALL      96
      IF (MOD1 .NE. 1) GO TO 90                                             WALL      97
      IF (L .EQ. 1) GO TO 80                                                WALL      98
      PXX=(2.*P(2,M)-P(3,M)-PW)*DDX                                         WALL      99
```

```
      AAXX=(2.*UNOR(2,IF)-UNOR(3,IF))*DDX                                WALL      100
      PZCOR(M)=ALAM*(XRW*PXX-DWW*AAXX/BETA)                              WALL      101
      GO TO 90                                                           WALL      102
   80 PZ=PZ+PZCOR(M)                                                     WALL      103
   90 IF (M .GT. ISWSMO) GO TO 100                                       WALL      104
      CALL RGAS(P(3,M),D(3,M),SW3,4)                                     WALL      105
      CALL RGAS(P(2,M),D(2,M),SW2,4)                                     WALL      106
      SZ=2.0*SW2-SW3 $ IF (SW2 .LT. SW3) SZ=.5*(SW2+SW3) $ GO TO 125     WALL      107
  100 SZ=-BB*DSY/WW                                                      WALL      108
  125 CONTINUE                                                           WALL      109
      PZ=PZ/PW                                                           WALL      110
      MOD1=KMOD1 $ ISWMOD=KSWMOD                                         WALL      111
      IF (IJUMP(M) .EQ. 1 .AND. L .EQ. 0) PZ=0.                          WALL      112
      BETA=BET                                                           WALL      113
      RETURN                                                             WALL      114
      END                                                                WALL      115
```

```
      SUBROUTINE SHOCK(M,JR,JL,JSG,IFF,L)                                SHOCK     2
C     SHOCK COMPUTES  PREDICTED OR CORRECTED Z DERIVATIVES OF C,C7,      SHOCK     3
C    AND CPHI.  JR AND JL ARE LINE INDEXES FOR Y DIFFS.                  SHOCK     4
C    JSG=1,2,3.  LINE INDEX FOR TRANG QUANTITIES                         SHOCK     5
C     IFF=1,2 LINE INDEX FOR CF AND CE AREAYS                            SHOCK     6
C     L=0  CORRESPINDS TO PREDICTOR                                      SHOCK     7
C     L=1  CORRESPONDS TO CORRECTOR                                      SHOCK     8
C     THIS VERSION HAD SECOND ORDER DIFFERENCING                         SHOCK     9
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                         NEWCOM    1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                NEWCOM    2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)     NEWCOM    3
      COMMON CU(4,20,25),CUP(4,20,25)                                    NEWCOM    4
C   ***   END OF BLANK COMMON   ***                                      CD3CSS   32
      COMMON /CSHOCK/ DCZ,DCZZ,DCPHZ,PDIF,SPDIF,DINX,D1INF,D2INF         CSHOCK    2
     1   ,UZCOR(4,25)                                                    CSHOCK    3
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                       CBODY     2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)            CBODY     3
      COMMON /BLK01/ CZY(3),CPHIY(3),V2(3),VOWY(3),PWY(3),SWY(3)         BLK01     2
     1   ,UNOR(3,2),CF(4,20,2),CG(4,20,3),CE(4,20,2)                     BLK01     3
      COMMON /BLK04/ GAMMA,GB,GD,GE,GA2,DDX,DDY,HOT2,ELIM,LCNT,ISWSMO,NA BLK04     2
     1   ,SW(25),GM(20,25)                                               BLK04     3
      COMMON/RGASS/AX,HX,TX,RRX,GX,NTEST,NGAS,NFIRST                     SHOCK    11
      DIMENSION DCUZ(4),CUZCOR(4)                                        SHOCK    12
      DATA(ISC1NEG=1)                                                    SHOCK    13
      CM=C(M)  $ CPHIM=CPHI(M)  $ CZM=CZ(M)  $ CPHIC=CPHIM/CM            SHOCK    14
      CMU1=1.+CPHIC**2  $ CMU2=CMU1+CZM**2                               SHOCK    15
      YPHIJ=YPHI(JSG)  $ YZJ=YZ(JSG)                                     SHOCK    16
      US=U(NC,M)*SPDIF $ WS=W(NC,M)*SPDIF $ PS=P(NC,M)/PINF              SHOCK    17
      VS=V(NC,M)*SPDIF $ $DS=D(NC,M)/DINF  $ ASQS=ASQ(NC,M)/PDIF         SHOCK    18
      WS2=WS*WS $ ETA=WS2-ASQS                                           SHOCK    19
      UV=US-CPHIC*VS                                                     DUMP     37
      CDUMP=(ETA*CMU1+UV**2)/ASQS                                        DUMP     38
      IF (CDUMP .GE. 0.) GO TO 1001                                      DUMP     39
      CALL DMPSQRT(5HSHOCK,1,Z,K,M,NC,CDUMP)                             DUMP     40
 1001 RETT=SQRT(CDUMP)                                                   DUMP     41
      ALAM=-ASQS*(RETT*WS+UV)/ETA                                        SHOCK    21
      IF(L.EQ.1) GO TO 20                                                SHOCK    22
      DO 10 I=1,4                                                        SHOCK    23
      CUZCOR(I)=0.                                                       SHOCK    24
   10 UZCOR(I,M)=CF(I,NC,IFF)+CF(I,NC-2,IFF)-2.*CF(I,NA,IFF)             SHOCK    25
      GO TO 30                                                           SHOCK    26
   20 DO 25 I=1,4                                                        SHOCK    27
   25 CUZCOR(I)=UZCOR(I,M)                                               SHOCK    28
   30 DO 35 I=1,4                                                        SHOCK    29
   35 DCUZ(I)=((CF(I,NC,IFF)-CF(I,NA,IFF)+CUZCOR(I))*DDX                 SHOCK    30
     1 +(CG(I,NC,JR)-CG(I,NC,JL))*DDY+CE(I,NC,IFF))/PINF                 SHOCK    31
      DCUZ(I)=DCUZ(I)/SPDIF                                              SHOCK    32
C *** THE EQUATION FOR XK1 IS VALID FOR PERFECTS GAS ONLY ***            SHOCK    33
      XK1=-DS/(ASQS*GB)                                                  SHOCK    34
      IF(NTEST.GE.0)GO TO 60                                             SHOCK    35
      PX=PINF*PS                                                         SHOCK    36
      DX=1.025*DS*DINF                                                   SHOCK    37
      CALL RGAS(PX,DX,DUMMY,4)                                           SHOCK    38
      HX1=HX                                                             SHOCK    39
      DX=.975*DS*DINF                                                    SHOCK    40
      CALL RGAS(PX,DX,DUMMY,4)                                           SHOCK    41
      XK1=.05*DS/(HX1-HX)*PINF/DINF                                      SHOCK    42
```

```
 60 CONTINUE                                                              SHOCK    43
    DUM1=XK1/DS  $ DUM=DUM1*ALAM                                          SHOCK    44
    RHS=(2.*ALAM-DUM*(WS2+US**2+VS**2))*DCUZ(1)+(UV-ALAM+DUM*WS2)         SHOCK    45
   1  *DCUZ(2)/WS+(DUM*US -1.)*DCUZ(3)+(CPHIC+DUM*VS)*DCUZ(4)             SHOCK    46
    VNIOMU=(DINX*C7M+D1INF*(COSPHI(M)+SINPHI(M)*CPHIC))/CMU2              SHOCK    47
   1    +D2INF*(SINPHI(M)-COSPHI(M)*CPHIC)/CMU2                          SHOCK    48
    DS2=DS**2  $ XMNSQ=(CMU2*VNIOMU**2)/(ASQS*DS2)                        SHOCK    49
    DUM2=1.-1./DS                                                         SHOCK    50
    A0=DUM2*(1.+XMNSQ+XK1*(PS-1.)/DS2)/(1.-XMNSQ)                         SHOCK    51
    DUM3=DINX-C7M*VNIOMU                                                  SHOCK    52
    AI=RETT*VNIOMU*(A0+DUM2)+DS*DUM3*A0                                   SHOCK    53
    DUM4=(PS-1.)/CMU2                                                     SHOCK    54
    C1=DUM3*AI-DUM4*CMU1                                                  SHOCK    55
    C2=(VS-CPHIC*VNIOMU/DS)*AI+CZM*CPHIC*DUM4                             SHOCK    56
    RHS1=RHS-(AI-DS*WS)*(VS+CPHIC*US)*YZJ/YPHIJ                           SHOCK    57
    IF(ISC1NEG.EQ.1) GO TO 40                                             SHOCK    58
    IF(C1*C1OLD.GT.0.) GO TO 40                                           SHOCK    59
    WRITE (6,1000) C1,L,M,K                                               SHOCK    60
1000 FORMAT(1H0,*IN SUBROUTINE SHOCK  C1 =*,1PE15.6,2X,*L,M,K =*,3I5)     SHOCK    61
    CALL SAVE(DUM,DUM,DUM)                                                SHOCK    62
 40 C1OLD=C1                                                              SHOCK    63
    ISC1NEG=0                                                             SHOCK    64
    DCZ=CZM-CPHIM*YZJ/YPHIJ                                               SHOCK    65
    DCPHZ=(YPHIJ*(CZY(JR)-CZY(JL))-YZJ*(CPHIY(JR)-CPHIY(JL)))*ODY         SHOCK    66
    DCZZ=(RHS1-C2*(DCPHZ-CPHIC*DCZ)/CM)/C1                                SHOCK    67
    RETURN                                                                SHOCK    68
    END                                                                   SHOCK    69
```

```
      SUBROUTINE INTEG(IFLAG)                                          INTEG     2
C                                                                      INTEG     3
C      INTEG INTEGRATES THE WALL PRESSURE TO OBTAIN THE                INTEG     4
C      FORCES AND MOMENTS (ABOUT ORIGIN) AND THEIR Z DERIVATIVES.      INTEG     5
C      THIS ROUTINE USED SIMPSON'S RULE FOR PHI INTEGRATION            INTEG     6
C      AND TRAPEZOIDAL RULE FOR Z INTEGRATION.                         INTEG     7
C                                                                      INTEG     8
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                       NEWCOM    1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)              NEWCOM    2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)   NEWCOM    3
      COMMON CU(4,20,25),CUP(4,20,25)                                  NEWCOM    4
C   ***   END OF BLANK COMMON   ***                                    CD3CSS   32
      COMMON /CINTEG/ FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ       CINTEG    2
     1   ,DYD3,MA,GY(25)                                               CINTEG    3
      REAL MX,MY,MZ,MXZ,MYZ,MZZ                                        CINTEG    4
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                     CBODY     2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)          CBODY     3
C                                                                      INTEG    10
      DIMENSION SUMJ(6,2),SF(6),SF1(6),SF2(6),F(6)                     INTEG    11
      IF (IDYAW .EQ. 1) GO TO 300                                      INTEG    12
C   ***   SIMPSON'S RULE FOR SYMMETRY CASE (PHIO=180)   ***            INTEG    13
      SUMJ(1,1)=SUMJ(2,1)=SUMJ(3,1)=SUMJ(1,2)=SUMJ(2,2)=SUMJ(3,2)=0.   INTEG    14
      K1=1                                                             INTEG    15
      DO 200 M=1,MA                                                    INTEG    16
      BM=B(M) $ BPHIB=BPHI(M)/BM                                       INTEG    17
      DUM=2.*(P(1,M)-PINF)*BM*PHIO*GY(M)                               INTEG    18
      SINP=SINPHI(M) $ COSP=COSPHI(M)                                  INTEG    19
      SF(1)=DUM*(COSP+BPHIB*SINP)                                      INTEG    20
      SF(3)=DUM*BZ(M) $ DUM1=SF(3)*BM                                  INTEG    21
      SF(2)=DUM1*COSP                                                  INTEG    22
      IF (M .NE. 2) GO TO 125                                          INTEG    23
      SF1(1)=SUMJ(1,1) $ SF1(2)=SUMJ(2,1) $ SF1(3)=SUMJ(3,1)           INTEG    24
  125 DO 150 I=1,3                                                     INTEG    25
  150 SUMJ(I,K1)=SUMJ(I,K1)+SF(I)                                      INTEG    26
      K1=3-K1                                                          INTEG    27
  200 CONTINUE                                                         INTEG    28
      SF2(1)=-2.*(P(1,MC)-PINF)*B(MC)*PHIO*GY(MC)                      INTEG    29
      SF2(3)=-SF2(1)*BZ(MC)                                            INTEG    30
      SF2(2)=-SF2(3)*B(MC)                                             INTEG    31
      DO 250 I=1,3                                                     INTEG    32
      F(I)=DYD3*(4.*SUMJ(I,2)+2.*SUMJ(I,1))                            INTEG    33
      IF (K1 .EQ. 1) GO TO 225                                         INTEG    34
      F(I)=F(I)+DYD3*(.5*SF(I)-SF1(I)+1.5*SF2(I))                      INTEG    35
      GO TO 250                                                        INTEG    36
  225 F(I)=F(I)+DYD3*(SF2(I)-SF1(I))                                   INTEG    37
  250 CONTINUE                                                         INTEG    38
      GO TO 600                                                        INTEG    39
C   ***   SIMPSON'S RULE FOR NON-SYMMETRIC CASE (PHIO=360)   ***       INTEG    40
  300 DO 325 I=1,6                                                     INTEG    41
      SUMJ(I,1)=0. $ SUMJ(I,2)=0.                                      INTEG    42
  325 CONTINUE                                                         INTEG    43
      K1=1                                                             INTEG    44
      DO 500 M=1,MA                                                    INTEG    45
      BM=B(M) $ BPHIB=BPHI(M)/BM                                       INTEG    46
      DUM=(P(1,M)-PINF)*BM*PHIO*GY(M)                                  INTEG    47
      SINP=SINPHI(M) $ COSP=COSPHI(M)                                  INTEG    48
      SF(1)=DUM*(COSP+BPHIB*SINP)                                      INTEG    49
```

```
      SF(3)=DUM*B7(M) $ DUM1=SF(3)*BM                                    INTEG     50
      SF(2)=DUM1*COSP $ SF(4)=DUM1*SINP                                  INTEG     51
      SF(5)=DUM*(BPHIB*COSP-SINP)                                        INTEG     52
      SF(6)=DUM*BM*BPHIB                                                 INTEG     53
      IF (M .NE. 2) GO TO 375                                            INTEG     54
      DO 350 I=1,6                                                       INTEG     55
      SF1(I)=SUMJ(I,1) $ SF2(I)=SF(I)                                    INTEG     56
  350 CONTINUE                                                           INTEG     57
  375 DO 400 I=1,6                                                       INTEG     58
  400 SUMJ(I,K1)=SUMJ(I,K1)+SF(I)                                        INTEG     59
      K1=3-K1                                                            INTEG     60
  500 CONTINUE                                                           INTEG     61
      DO 525 I=1,6                                                       INTEG     62
      F(I)=DYD3*(4.*SUMJ(I,2)+2.*SUMJ(I,1))                              INTEG     63
      IF (K1 .EQ. 1) GO TO 525                                           INTEG     64
      F(I)=F(I)+DYD3*(4.*SF1(I)-SF2(I)-2.*SF(I))                         INTEG     65
  525 CONTINUE                                                           INTEG     66
C                                                                        INTEG     67
  600 F(2)=F(2)+Z*F(1)                                                   INTEG     68
      IF (IDYAW .EQ. 0) GO TO 625                                        INTEG     69
      F(4)=F(4)-Z*F(5) $ F(6)=-F(6)                                      INTEG     70
  625 IF (IFLAG .EQ. 0) GO TO 700                                        INTEG     71
      HH=(Z-ZP)/2.                                                       INTEG     72
      FN=FN+HH*(F(1)+FNZ)                                                INTEG     73
      MY=MY+HH*(F(2)+MYZ)                                                INTEG     74
      FA=FA+HH*(F(3)+FAZ)                                                INTEG     75
      IF (IDYAW .EQ. 0) GO TO 700                                        INTEG     76
      MX=MX+HH*(F(4)+MXZ)                                                INTEG     77
      FY=FY+HH*(F(5)+FYZ)                                                INTEG     78
      MZ=MZ+HH*(F(6)+MZZ)                                                INTEG     79
  700 ZP=Z                                                               INTEG     80
      FNZ=F(1) $ MYZ=F(2) $ FAZ=F(3)                                     INTEG     81
      IF (IDYAW .EQ. 0) GO TO 800                                        INTEG     82
      MXZ=F(4) $ FYZ=F(5) $ MZZ=F(6)                                     INTEG     83
  800 RETURN                                                             INTEG     84
      END                                                                INTEG     85
```

```
      SUBROUTINE INTRPL(L,X,Y,N,XX,YY)                                  INTRPL     2
C                                                                       INTRPL     3
C      THIS IS A LINEAR INTERPOLATION ROUTINE.  THE DATA IN THE X AND   INTRPL     4
C      XX ARRAYS ARE ASSUMED TO BE INCREASING AND NO CHECKING IS DONE.  INTRPL     5
C                                                                       INTRPL     6
      DIMENSION X(L),Y(L),XX(N),YY(N)                                   INTRPL     7
      M=2                                                               INTRPL     8
      DO 200 I=1,N                                                      INTRPL     9
      DO 100 J=M,L                                                      INTRPL    10
      IF (XX(I) .LE. X(J)) GO TO 150                                    INTRPL    11
  100 CONTINUE                                                          INTRPL    12
      J=L                                                               INTRPL    13
  150 M=J                                                               INTRPL    14
      YY(I)=Y(J-1)+(Y(J)-Y(J-1))*(XX(I)-X(J-1))/(X(J)-X(J-1))           INTRPL    15
  200 CONTINUE                                                          INTRPL    16
      RETURN                                                            INTRPL    17
      END                                                               INTRPL    18
```

```
      SUBROUTINE REZONE(NCNEW,MCNEW,ROLD,PHIOLD,DCUB,                    REZONE     2
     1   DARR1,DARR2,DARR3,DARR4,NDIM,MDIM)                              REZONE     3
C                                                                        REZONE     4
C      REZONE TAKES THE INITIAL DATA AND REZONES THE MESH                REZONE     5
C                                                                        REZONE     6
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                         NEWCOM     1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                NEWCOM     2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)     NEWCOM     3
      COMMON CU(4,20,25),CUP(4,20,25)                                    NEWCOM     4
C   ***   END OF BLANK COMMON   ***                                      CD3CSS    32
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                       CBODY      2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)            CBODY      3
      COMMON /BLK02/ THETA,DY,TG4(3),TG5(3),TG6(3)                       BLK02      2
     1   ,X(20),XZ(20,2),XR(20,2),XPHI(20,2),Y(25)                       BLK02      3
     2   ,TF4(20,2),TF6(20,2),TF7(20,2)                                  BLK02      4
C                                                                        REZONE     8
      DIMENSION ROLD(NDIM,MDIM),DCUB(NDIM,MDIM,4),DARR4(NDIM)            REZONE     9
      DIMENSION PHIOLD(MDIM),DARR1(MDIM),DARR2(MDIM),DARR3(MDIM)         REZONE    10
C                                                                        REZONE    11
C      NOTE THAT THE WAY DCUB IS USED IN THIS ROUTINE                    REZONE    12
C      REQUIRES THAT D,P,U,V ARE CONSECUTIVE IN COMMON                   REZONE    13
C                                                                        REZONE    14
      NCOLD=NC $ MCOLD=MC $ NC=NCNEW $ MC=MCNEW                          REZONE    15
      DDX=NC-1 $ DDY=MC-1                                                REZONE    16
      DO 25 M=1,MCOLD                                                    REZONE    17
      PHIOLD(M)=PHI(M)                                                   REZONE    18
      DO 20 N=1,NCOLD                                                    REZONE    19
      ROLD(N,M)=R(N,M)                                                   REZONE    20
   20 CONTINUE                                                           REZONE    21
   25 CONTINUE                                                           REZONE    22
      DO 50 N=1,NC                                                       REZONE    23
      X(N)=(N-1)/DDX                                                     REZONE    24
   50 CONTINUE                                                           REZONE    25
      DO 75 M=1,MC                                                       REZONE    26
      Y(M)=(M-1)/DDY                                                     REZONE    27
      CALL TRANG(Y(M),M,1) $ PHI(M)=THETA*PHIO                           REZONE    28
      COSPHI(M)=COS(PHI(M)) $ SINPHI(M)=SIN(PHI(M))                      REZONE    29
      CALL BODY(M)                                                       REZONE    30
   75 CONTINUE                                                           REZONE    31
      CALL INTRPL(MCOLD,PHIOLD,C   ,MC,PHI,DARR1)                        REZONE    32
      CALL INTRPL(MCOLD,PHIOLD,CZ  ,MC,PHI,DARR2)                        REZONE    33
      CALL INTRPL(MCOLD,PHIOLD,CPHI,MC,PHI,DARR3)                        REZONE    34
      DO 80 M=1,MC                                                       REZONE    35
      C(M)=DARR1(M)                                                      REZONE    36
      CZ(M)=DARR2(M)                                                     REZONE    37
      CPHI(M)=DARR3(M)                                                   REZONE    38
   80 CONTINUE                                                           REZONE    39
      DO 85 M=1,MC                                                       REZONE    40
      CALL TRANF(M,1,1)                                                  REZONE    41
   85 CONTINUE                                                           REZONE    42
      DO 225 N=1,NCOLD                                                   REZONE    43
      DO 100 M=1,MCOLD                                                   REZONE    44
      DARR1(M)=ROLD(N,M)                                                 REZONE    45
  100 CONTINUE                                                           REZONE    46
      CALL INTRPL(MCOLD,PHIOLD,DARR1,MC,PHI,DARR2)                       REZONE    47
      DO 125 M=1,MC                                                      REZONE    48
      ROLD(N,M)=DARR2(M)                                                 REZONE    49
```

```
125 CONTINUE                                                          REZONE    50
    DO 200 I=1,4                                                      REZONE    51
    DO 150 M=1,MCOLD                                                  REZONE    52
    DARR1(M)=DCUB(N,M,I)                                              REZONE    53
150 CONTINUE                                                          REZONE    54
    CALL INTRPL(MCOLD,PHIOLD,DARR1,MC,PHI,DARR2)                      REZONE    55
    DO 175 M=1,MC                                                     REZONE    56
    DCUB(N,M,I)=DARR2(M)                                              REZONE    57
175 CONTINUE                                                          REZONE    58
200 CONTINUE                                                          REZONE    59
225 CONTINUE                                                          REZONE    60
    DO 300 I=1,4                                                      REZONE    61
    DO 275 M=1,MC                                                     REZONE    62
    CALL INTRPL(NCOLD,ROLD(1,M),DCUB(1,M,I),NC,R(1,M),DARR4)          REZONE    63
    DO 250 N=1,NC                                                     REZONE    64
    DCUB(N,M,I)=DARR4(N)                                              REZONE    65
250 CONTINUE                                                          REZONE    66
275 CONTINUE                                                          REZONE    67
300 CONTINUE                                                          REZONE    68
    RETURN                                                            REZONE    69
    END                                                               REZONE    70
```

```
      SUBROUTINE RGAS(PX,RX,SX,NUMX)                                    RGAS      2
CRGASR  WALKER  TEMP CONVERTED TO RANKINE                               RGAS      3
      COMMON/RGASS/AX,HX,TX,RRX,GX,NTEST,NGAS,NFIRST                    RGAS      4
      DIMENSION TH(5,600),NDL(4,11),NDU(4,11),AN(4),C(7),ANR(17),BN(4)  RGAS      5
      COMMON/CSERCH/F,TH5(600),NL,NU,NOUT,NER                           RGAS      6
      COMMON/SAVRG/PO,RO,B,D,E,FM,NDL,NDU,MIDL,TH,RTO,SQPORO,CAX,CHX     RGAS      7
      DIMENSION MIDL(4,11), SAVF0(4)                                    RGAS      8
      DATA IF0 /0/                                                      RGAS      9
C.....THE ARRAYS NDL AND NDU MUST BE STORED IN ADJACENT LOCATIONS.       RGAS     10
C                                                                       RGAS     11
C                                                                       RGAS     12
      P=PX                                                              RGAS     13
      S=SX                                                              RGAS     14
      R=RX                                                              RGAS     15
      NUM=NUMX                                                          RGAS     16
      NUM5=NUM-5                                                        RGAS     17
      IF(NFIRST-NGAS)10,9,10                                            RGAS     18
   10 NFIRST=NGAS                                                       RGAS     19
      IF(NTEST)7,19,19                                                  RGAS     20
C.....LOAD PERFECT GAS CONSTANTS.                                       RGAS     21
   19 ANR(1)=RRX                                                        RGAS     22
      ANR(2)=GX                                                         RGAS     23
      ANR(3)=ANR(1)/(ANR(2)-1.)                                         RGAS     24
      ANR(4)=ANR(1)+ANR(3)                                              RGAS     25
      ANR(5)=1./ANR(2)                                                  RGAS     26
      ANR(6)=ANR(4)/ANR(1)                                              RGAS     27
      ANR(7)=ANR(6)/ANR(2)                                              RGAS     28
      ANR(8)=49008.609-ANR(3)*ALOG(171.6/.0001**ANR(2))                 RGAS     29
      GO TO 8                                                           RGAS     30
C.....LOAD REAL GAS CONSTANTS.                                          RGAS     31
    7 CONTINUE                                                          RGAS     32
      CALL LOCATE(NGAS,9)                                               RGAS     33
      READ(9)(NDL(N),N=1,89)                                            RGAS     34
      NMM=NDL(89)                                                       RGAS     35
      READ(9) (TH(N),N=1,NMM),WTMIX,(C(N),N=1,7)                        RGAS     36
      REWIND 9                                                          RGAS     37
      DO 115  N=1,44                                                    RGAS     38
  115 MIDL(N)=(NDL(N)+NDU(N))/2                                         RGAS     39
      NDUMX=NDU(44)                                                     RGAS     40
      DO 120  N=1,NDUMX                                                 RGAS     41
  120 TH5(N)=TH(5,N)                                                    RGAS     42
      F=0.                                                              RGAS     43
      DO 21  N1=1,4                                                     RGAS     44
      NL=NDL(N1,1)                                                      RGAS     45
      NU=NDU(N1,1)                                                      RGAS     46
      NER=MIDL(N1,1)                                                    RGAS     47
      CALL SERCH                                                        RGAS     48
   21 SAVF0(N1)=TH(1,NOUT)                                              RGAS     49
      CONC=WTMIX/28.966                                                 RGAS     50
      PO=2116.                                                          RGAS     51
      PO=.002498*CONC                                                   RGAS     52
      RRR=1716./CONC                                                    RGAS     53
      RRX=RRR                                                           RGAS     54
      RTO=RRR*493.635                                                   RGAS     55
      SQPORO=SQRT(RO/PO)                                                RGAS     56
      B=TH(NMM-2)                                                       RGAS     57
      E=TH(NMM-1)                                                       RGAS     58
```

```
      D=TH(NMM)                                                    RGAS      59
      FM=2.1632+.3468*CONC                                         RGAS      60
      AA=D*FM                                                      RGAS      61
      BB=E*FM+1.                                                   RGAS      62
      CCC=B+FM                                                     RGAS      63
      RLAST=0.0                                                    RGAS      64
      RSTART=100.0                                                 RGAS      65
      Z2=RO/10.**7                                                 RGAS      66
      PR=-7.+R                                                     RGAS      67
      PR=PO*10.**PR                                                RGAS      68
      Z1=PR                                                        RGAS      69
      IF0=IF0+1                                                    RGAS      70
      F=0.0                                                        RGAS      71
      DO 23  N1=1,4                                                RGAS      72
   23 BN(N1)=SAVF0(N1)                                             RGAS      73
      BN(1)=BN(1)/SQPORO                                           RGAS      74
      BN(2)=BN(2)*RTO                                              RGAS      75
      BN(3)=BN(3)*1.8                                              RGAS      76
      BN(4)=BN(4)*RRR                                              RGAS      77
      ANR(9)=PR/(Z2*BN(3))                                         RGAS      78
      RRX=ANR(9)                                                   RGAS      79
      ANR(12)=BN(2)/BN(3)                                          RGAS      80
      ANR(10)=1.+ANR(9)/(ANR(12)-ANR(9))                           RGAS      81
      ANR(11)=ANR(12)/ANR(10)                                      RGAS      82
      ANR(13)=1./ANR(10)                                           RGAS      83
      ANR(14)=ANR(12)/ANR(9)                                       RGAS      84
      ANR(16)=BN(4)-ANR(11)*ALOG(Z1/Z2**ANR(10))                   RGAS      85
      ANR(17)=BN(1)*BN(1)*Z2/Z1                                    RGAS      86
      CHX=ANR(14)                                                  RGAS      87
      CAX=ANR(17)                                                  RGAS      88
    9 IF(NTEST)16,8,8                                              RGAS      89
C.....CALCULATE F AND NR. INITIALIZE CONTROL INTEGERS              RGAS      90
   16 P=ALOG(P/PO)/2.3025851                                       RGAS      91
      IF(NUM5)40,31,70                                             RGAS      92
   31 REAL=S/RRR                                                   RGAS      93
      GG=(REAL-C(1)-C(2)*P)/(C(3)+P*(C(4)+P*C(5)))                 RGAS      94
      R=C(6)*GG+C(7)*P                                             RGAS      95
      PER=ABS((RSTART-R)/R)                                        RGAS      96
      RSTART=R                                                     RGAS      97
      IF(PER.LT.0.1) R=RLAST                                       RGAS      98
      RL=R-8                                                       RGAS      99
      CC=CCC-P                                                     RGAS     100
      RH=-CC*(1.+AA*CC/(BB*BB))/BB+.005                            RGAS     101
      IF(RH.LT.-7.) RH=-7.0                                        RGAS     102
      IF(R.LT.RH) R=RH                                             RGAS     103
      IF(RL.GT.3.0) RL=3.0                                         RGAS     104
      IF(RL.LT.R) R=RL                                             RGAS     105
      NUMR=0                                                       RGAS     106
      NIMX=0                                                       RGAS     107
      NUMM=5                                                       RGAS     108
      NUMM9P=NUMM-9+NUM                                            RGAS     109
      NBOT=4                                                       RGAS     110
      NUP=4                                                        RGAS     111
      GO TO 42                                                     RGAS     112
   40 R=ALOG(R/RO)/2.3025851                                       RGAS     113
      NUMM=5                                                       RGAS     114
      NUMM9P=NUMM-9+NUM                                            RGAS     115
```

```
      NUP=NUM                                                          RGAS      116
      NBOT=1                                                           RGAS      117
   42 CONTINUE                                                         RGAS      118
      IF (R) 11,12,13                                                  RGAS      119
   11 NR=R-1.0                                                         RGAS      120
      IF(NR.LT.-7) NR=-7                                               RGAS      121
      GO TO 15                                                         RGAS      122
   12 NR=-1.0                                                          RGAS      123
      GO TO 15                                                         RGAS      124
   13 NR=R                                                             RGAS      125
      IF(NR.GT.2) NR=2                                                 RGAS      126
   15 DX=R-FLOAT(NR)                                                   RGAS      127
      NR=NR+8                                                          RGAS      128
      F=(P-R-A)/(1.+R*(E+D*R))                                         RGAS      129
      IF(F) 158,160,160                                                RGAS      130
  158 IF(F.GT.-1.0E-08) F=0.0                                          RGAS      131
  160 CONTINUE                                                         RGAS      132
C                   IF(NUMM-9+NUM) 22,162,22      ----------IF(NUMM9P)  RGAS      133
      IF(NUMM9P) 22,162,22                                             RGAS      134
  162 IF(F-.000001) 27,161,161                                         RGAS      135
  161 IF(FM.LT.F) GO TO 44                                             RGAS      136
C.....SEARCH FOR CORRECT COEFFICIENTS AND CALCULATE DESIRED PROPERTIES. RGAS      137
   22 DO 17  N1=NBOT,NUP                                               RGAS      138
      NL=NDL(N1,NR)                                                    RGAS      139
      NU=NDU(N1,NR)                                                    RGAS      140
      NER=MIDL(N1,NR)                                                  RGAS      141
      CALL SERCH                                                       RGAS      142
      Y1=TH(1,NOUT)+F*(TH(2,NOUT)+F*(TH(3,NOUT)+F*TH(4,NOUT)))         RGAS      143
      NL=NDL(N1,NR+1)                                                  RGAS      144
      NU=NDU(N1,NR+1)                                                  RGAS      145
      NER=MIDL(N1,NR+1)                                                RGAS      146
      CALL SERCH                                                       RGAS      147
      Y2=TH(1,NOUT)+F*(TH(2,NOUT)+F*(TH(3,NOUT)+F*TH(4,NOUT)))         RGAS      148
      AN(N1)=Y1+DX*(Y2-Y1)                                             RGAS      149
   17 CONTINUE                                                         RGAS      150
      IF(NUM5) 51,52,52                                                RGAS      151
   52 IF(NUMM9P) 39,108,39                                             RGAS      152
  108 RX=RO*10.**R                                                     RGAS      153
   51 GO TO (121,122,123,124,124),NUM                                  RGAS      154
C.....NORMALIZE FINAL QUANTITIES.                                      RGAS      155
  124 SX=AN(4)*RRR                                                     RGAS      156
  123 TX=AN(3)*1.8                                                     RGAS      157
  122 HX=AN(2)*RTO                                                     RGAS      158
  121 AX=AN(1)/SQPORO                                                  RGAS      159
      GO TO 109                                                        RGAS      160
C.....ENTROPY INTERATION                                               RGAS      161
   39 DIFF=ABS((REAL-AN(NUP))/REAL)                                    RGAS      162
      IF(DIFF-.0001) 37,37,38                                          RGAS      163
C                     NUMM9P=NUMM-9+NUM=0                              RGAS      164
   37 NUMM9P=0                                                         RGAS      165
      NUMM=9-NUM                                                       RGAS      166
      NBOT=1                                                           RGAS      167
      NUP=3                                                            RGAS      168
      AN(4)=REAL                                                       RGAS      169
      RLAST=R                                                          RGAS      170
      GO TO 42                                                         RGAS      171
   38 NUMB=NUMB+1                                                      RGAS      172
```

```
      NIMX=NIMX+1                                                       RGAS  173
      IF(NIMX.GT.20) GO TO 44                                           RGAS  174
   43 IF(NUMB-2) 82,83,84                                               RGAS  175
   82 IF(REAL-AN(NUP)) 85,37,86                                         RGAS  176
   85 R1=R                                                              RGAS  177
      S1=AN(NUP)                                                        RGAS  178
      R=R+.3                                                            RGAS  179
      IF(RL.LT.R) R=RL                                                  RGAS  180
      R2=R                                                              RGAS  181
      L=0                                                               RGAS  182
      GO TO 42                                                          RGAS  183
   86 R2=R                                                              RGAS  184
      S2=AN(NUP)                                                        RGAS  185
      R=R-.3                                                            RGAS  186
      IF(R.LT.RH) R=RH                                                  RGAS  187
      R1=R                                                              RGAS  188
      L=1                                                               RGAS  189
      GO TO 42                                                          RGAS  190
   83 IF(L) 91,90,91                                                    RGAS  191
   90 S2=AN(NUP)                                                        RGAS  192
      R=R2                                                              RGAS  193
      IF(S1.NE.S2) R=R2-(S2-REAL)/(S2-S1)*(R2-R1)                       RGAS  194
      IF(RL.LT.R) R=RL                                                  RGAS  195
      GO TO 93                                                          RGAS  196
   91 S1=AN(NUP)                                                        RGAS  197
      R=(REAL-S1)/(S2-S1)*(R2-R1)+R1                                    RGAS  198
      IF(R.LT.RH) R=RH                                                  RGAS  199
   93 IF(R2-R) 104,37,105                                               RGAS  200
  104 NUMB=1                                                            RGAS  201
      R1=R2                                                             RGAS  202
      S1=S2                                                             RGAS  203
      L=0                                                               RGAS  204
      IF(R2+.3-RL) 210,211,211                                          RGAS  205
  211 R2=RL                                                             RGAS  206
      R=R2                                                              RGAS  207
      GO TO 42                                                          RGAS  208
  210 R2=R2+.3                                                          RGAS  209
      R=R2                                                              RGAS  210
      GO TO 42                                                          RGAS  211
  105 IF(R-R1) 106,37,42                                                RGAS  212
  106 NUMB=1                                                            RGAS  213
      R2=R1                                                             RGAS  214
      S2=S1                                                             RGAS  215
      L=1                                                               RGAS  216
      IF(RH-R1+.3) 212,213,213                                          RGAS  217
  213 R1=RH                                                             RGAS  218
      R=R1                                                              RGAS  219
      GO TO 42                                                          RGAS  220
  212 R1=R1-.3                                                          RGAS  221
      R=R1                                                              RGAS  222
      GO TO 42                                                          RGAS  223
   84 IF(REAL-AN(NUP)) 87,87,88                                         RGAS  224
   87 R1=R                                                              RGAS  225
      GO TO 91                                                          RGAS  226
   88 R2=R                                                              RGAS  227
      GO TO 90                                                          RGAS  228
   44 IF(F-.000001) 27,444,444                                          RGAS  229
```

```
C.....OUTSIDE GAS TABLES.                                          RGAS      230
  444 NTIMES=NTIMES+1                                              RGAS      231
      WRITE(6,190)                                                 RGAS      232
  190 FORMAT(1H0,10X,36HOUTSIDE TABLES IN RGAS ENTERING WITH)      RGAS      233
      WRITE(6,191) PX                                              RGAS      234
  191 FORMAT(11X,2HP=,E13.6)                                       RGAS      235
      IF(NUMS) 192,193,193                                         RGAS      236
  192 CONTINUE                                                     RGAS      237
      WRITE(6,194) RX                                              RGAS      238
  194 FORMAT(11X,2HR=E13.6)                                        RGAS      239
      GO TO 196                                                    RGAS      240
  193 CONTINUE                                                     RGAS      241
      WRITE(6,195) SX                                              RGAS      242
  195 FORMAT(11X,2HS=,E13.6)                                       RGAS      243
  196 IF(NTIMES-10) 109,197,197                                    RGAS      244
  197 CONTINUE                                                     RGAS      245
      WRITE(6,198)                                                 RGAS      246
  198 FORMAT(20X,28HEXIT CALLED ON TENTH FAILURE)                  RGAS      247
      GO TO 25                                                     RGAS      248
   70 Z=Z*SQRT(-1.0)                                               RGAS      249
   25 CALL EXIT                                                    RGAS      250
C.....REAL GAS BUT BELOW GAS TABLES.                               RGAS      251
   27 L=8                                                          RGAS      252
      P=PX                                                         RGAS      253
      R=RX                                                         RGAS      254
      L1=9                                                         RGAS      255
      GO TO 26                                                     RGAS      256
C.....PERFECT GAS                                                  RGAS      257
    8 L=0                                                          RGAS      258
      L1=2                                                         RGAS      259
   26 CONTINUE                                                     RGAS      260
      IF(NUMS)440,69,70                                            RGAS      261
  440 QUOT=P/R                                                     RGAS      262
      GO TO (65,66,67,68,69),NUM                                   RGAS      263
   69 EX=S-ANR(L+8)                                                RGAS      264
      EX=EXP(EX/ANR(L+3))                                          RGAS      265
      R=(P/EX)**ANR(L+5)                                           RGAS      266
      QUOT=P/R                                                     RGAS      267
      GO TO 67                                                     RGAS      268
   68 S=ANR(L+8)+ANR(L+3)*(ALOG(P)-ANR(L+2)*ALOG(R))               RGAS      269
   67 T=QUOT/ANR(L+1)                                              RGAS      270
   66 H=QUOT*ANR(L+6)                                              RGAS      271
   65 LL=L+L1                                                      RGAS      272
      A=SQRT(ANR(LL)*QUOT)                                         RGAS      273
   30 AX=A                                                         RGAS      274
      HX=H                                                         RGAS      275
      TX=T                                                         RGAS      276
      SX=S                                                         RGAS      277
      RX=R                                                         RGAS      278
  109 CONTINUE                                                     RGAS      279
      RETURN                                                       RGAS      280
      END                                                          RGAS      281
```

```
      SUBROUTINE HRGAS(PX,RX,QX,N1)                                    HRGAS     2
      DIMENSION NDL(4,11),NDU(4,11),MIDL(4,11),TH(5,600)               HRGAS     3
      COMMON/SAVRG/PO,RO,R,D,E,FM,NDL,NDU,MIDL,TH,RTO,SQPORO,CAX,CHX   HRGAS     4
      COMMON/RGASS/AX,HX,TX,RRX,GX,NTEST,NGAS                          HRGAS     5
      COMMON/CSERCH/F,TH5(600),NL,NU,NOUT,NER                          HRGAS     6
      P=ALOG(PX/PO)/2.3025851                                          HRGAS     7
      R=ALOG(RX/RO)/2.3025851                                          HRGAS     8
      IF(R)11,12,13                                                    HRGAS     9
   11 NR=R-1.0                                                         HRGAS    10
      IF(NR.LT.-7)NR=-7                                                HRGAS    11
      GO TO 15                                                         HRGAS    12
   12 NR=-1.                                                           HRGAS    13
      GO TO 15                                                         HRGAS    14
   13 NR=R                                                             HRGAS    15
      IF(NR.GT.2)NR=2                                                  HRGAS    16
   15 DX=R-FLOAT(NR)                                                   HRGAS    17
      NR=NR+8                                                          HRGAS    18
      F=(P-R-R)/(1.+R*(E+D*R))                                         HRGAS    19
      ENTRY ARGAS                                                      HRGAS    20
      IF(F)158,160,160                                                 HRGAS    21
  158 IF(F.GT.-1.0E-08)F=0.0                                           HRGAS    22
  160 IF(F-.000001)27,161,161                                          HRGAS    23
  161 IF(FM.LT.F)GO TO 44                                              HRGAS    24
      NL=NDL(N1,NR)                                                    HRGAS    25
      NU=NDU(N1,NR)                                                    HRGAS    26
      NER=MIDL(N1,NR)                                                  HRGAS    27
      CALL SERCH                                                       HRGAS    28
      Y1=TH(1,NOUT)+F*(TH(2,NOUT)+F*(TH(3,NOUT)+F*TH(4,NOUT)))         HRGAS    29
      NL=NDL(N1,NR+1)                                                  HRGAS    30
      NU=NDU(N1,NR+1)                                                  HRGAS    31
      NER=MIDL(N1,NR+1)                                                HRGAS    32
      CALL SERCH                                                       HRGAS    33
      Y2=TH(1,NOUT)+F*(TH(2,NOUT)+F*(TH(3,NOUT)+F*TH(4,NOUT)))         HRGAS    34
      HX=RTO*(Y1+DX*(Y2-Y1))                                           HRGAS    35
      IF(N1.EQ.2)RETURN                                                HRGAS    36
      QX=HX/(RTO*SQPORO)                                               HRGAS    37
      QX=QX*QX                                                         HRGAS    38
      RETURN                                                           HRGAS    39
   27 IF(N1.EQ.1)GO TO 28                                              HRGAS    40
      HX=CHX*PX/RX                                                     HRGAS    41
      RETURN                                                           HRGAS    42
   28 QX=CAX*PX/RX                                                     HRGAS    43
      RETURN                                                           HRGAS    44
   44 WRITE(6,190)                                                     HRGAS    45
  190 FORMAT(1H1,*OUTSIDE GAS TABLES*)                                 HRGAS    46
      STOP                                                             HRGAS    47
      END                                                              HRGAS    48
```

```
      SUBROUTINE SERCH                                          SERCH      2
      COMMON/CSERCH/X,Q(600),NL,NU,NOUT,NER                     SERCH      3
      NOUT=70000                                                SERCH      4
      IF(X.GE.Q(NER)) NL=NER                                    SERCH      5
      DO 10  I=NL,NU                                            SERCH      6
      IF(X.LT.Q(I)) RETURN                                      SERCH      7
      NOUT=I                                                    SERCH      8
   10 CONTINUE                                                  SERCH      9
      RETURN                                                    SERCH     10
       END                                                      SERCH     11
```

```
      SUBROUTINE LOCATE(NF,NT)                                    LOCATE     2
C                                                                 LOCATE     3
C      NF IS THE NUMBER OF THE FILE                               LOCATE     4
C      NT IS THE NUMBER OF THE TAPE                               LOCATE     5
C                                                                 LOCATE     6
      REWIND NT                                                   LOCATE     7
      IF (NF .LE. 1) GO TO 999                                    LOCATE     8
      IF=NF-1                                                     LOCATE     9
      DO 50 I=1,IF                                                LOCATE    10
   25 READ (NT) DUM                                               LOCATE    11
      IF (EOF(NT)) 50,25                                          LOCATE    12
   50 CONTINUE                                                    LOCATE    13
  999 RETURN                                                      LOCATE    14
      END                                                         LOCATE    15
```

```
      SUBROUTINE SHFAX(I,NDIM,MDIM,RN,UN,VN,WN,PN,DN,CPP,CZO,CON,CN,CNO,  SHFAX      2
     1 CPHIO)                                                             SHFAX      3
C                                                                         SHFAX      4
C       THIS ROUTINE SHIFTS Z AXES BY A PARALLEL DISPLACEMENT OF ZAS      SHFAX      5
C                                                                         SHFAX      6
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,PAD                          NEWCOM     1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                 NEWCOM     2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)      NEWCOM     3
      COMMON CU(4,20,25),CUP(4,20,25)                                     NEWCOM     4
C    ***    END OF BLANK COMMON    ***                                    CD3CSS    32
      COMMON /CBENT/ ZBB(25),ALNS,DST,AUQN,BETA,BSN,CENUF,DELII,DELTA     NEWCOM     5
     1    ,COSBN,EPSQ,ZMAXS,PID2,TANBN,IBN,HN,THETABN                     CBENT      3
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                        CBODY      2
     1    ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)            CBODY      3
      COMMON /CINTEG/ FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ          CINTEG     2
     1    ,DYD3,MA,GY(25)                                                 CINTEG     3
      REAL MX,MY,MZ,MXZ,MYZ,MZZ                                           CINTEG     4
      COMMON /BLK04/ GAMMA,GB,GD,GE,GA2,DDX,DDY,HOT2,ELIM,LCNT,ISWSMO,NA  BLK04      2
     1    ,SW(25),GM(20,25)                                               BLK04      3
      DIMENSION RN(NDIM,MDIM),UN(NDIM,MDIM),VN(NDIM,MDIM)                 SHFAX      9
     1   ,WN(NDIM,MDIM),PN(NDIM,MDIM),DN(NDIM,MDIM)                      SHFAX     10
      DIMENSION CPP(1),CZO(1),CON(1),CN(1),CNO(1),CPHIO(1)                SHFAX     11
      NA=NC-1  $ MA=MC-1                                                  SHFAX     12
      GO TO (10,11),I                                                     SHFAX     13
   10 ZAS=AMIN1(ZMAXS,HN)  $ HN=HN-ZAS                                    SHFAX     14
      DELII=DELII+ZAS  $ GO TO 12                                         SHFAX     15
   11 ZAS=HN  $ HN=0  $ DELII=DELII+ZAS                                   SHFAX     16
   12 SHIFDEL=ZAS*(COSBN**2-BSN**2)/AUQN                                  SHFAX     17
      DELTA=DELTA+SHIFDEL                                                 SHFAX     18
      DO 1 M=1,MC                                                         SHFAX     19
      CPP(M)=ATAN2(C(M)*SINPHI(M),C(M)*COSPHI(M)-ZAS)                     SHFAX     20
      IF (CPP(M) .LT.-1.E-8) CPP(M)=CPP(M)+2.*PI                          SHFAX     21
    1 CNO(M)= SQRT(C(M)**2-2.*C(M)*COSPHI(M)*ZAS+ZAS*ZAS)                 SHFAX     22
      CPP(1)=0. $ CPP(MC)=PHIO                                            SHFAX     23
      CALL INTRPL(MC,CPP,C,MC,PHI,CON)                                    SHFAX     24
      CALL INTRPL(MC,CPP,CNO,MC,PHI,CN)                                   SHFAX     25
      CALL INTRPL(MC,CPP,CZ,MC,PHI,CZO)                                   SHFAX     26
      CALL INTRPL(MC,CPP,CPHI,MC,PHI,CPHIO)                               SHFAX     27
      DO 2 M=1,MC                                                         SHFAX     28
      CA=ATAN2(CN(M)*SINPHI(M),CN(M)*COSPHI(M)+ZAS)                       SHFAX1     1
      CP2=SIN(CA)  $ CA=COS(CA)                                           SHFAX     30
      SCR=CA+CP2*CPHIO(M)/CON(M)                                          SHFAX1     2
      TCR=CA*CPHIO(M)/CON(M)-CP2                                          SHFAX1     3
      UCR=SCR*COSPHI(M)-TCR*SINPHI(M)                                     SHFAX1     4
      CZ(M)=CZO(M)/UCR                                                    SHFAX1     5
      CPHI(M)=CN(M)*(SCR*SINPHI(M)+TCR*COSPHI(M))/UCR                     SHFAX1     6
      S1=EPSQ*SINPHI(M)**2+1.  $ SCA=COSPHI(M)*DELTA                      SHFAX     36
      R(1,M)=(SQRT(SCA*SCA+(BETA+DELTA)*(BETA-DELTA)*S1)-SCA)/S1          SHFAX     37
      R (NC,M)=CN(M)                                                      SHFAX     38
      RDIF=(R (NC,M)-R (1,M))/NA                                          SHFAX     39
      DO 3 N=2,NA                                                         SHFAX     40
    3 R(N,M)=R(1,M)+(FLOAT(N)-1.)*RDIF                                    SHFAX     41
    2 CONTINUE                                                            SHFAX     42
      DELTA=DELTA-SHIFDEL                                                 SHFAX     43
      DO 20 M=1,MC                                                        SHFAX     44
      PHN=PHI(M)                                                          SHFAX     45
      DO 20 N=1,NC                                                        SHFAX     46
```

```
      RNC=R(N,M)                                                        SHFAX     47
      ROC=SQRT(RNC**2+2.*RNC*COSPHI(M)*ZAS+ZAS*ZAS)                     SHFAX     48
      PHO=ATAN2(RNC*SINPHI(M),RNC*COSPHI(M)+ZAS)                        SHFAX     49
      IF (PHO .LT.-1.E-8) PHO=PHO+2.*PI                                 SHFAX     50
      IF (M .EQ. MC) PHO=PHIO                                           SHFAX     51
      SINPHO=SIN(PHO) $ COSPHO=COS(PHO)                                 SHFAX1     7
      S1=EPSQ*SINPHO**2+1. $ SCA=COSPHO*DELTA                           SHFAX1     8
      BOD   =(SQRT(SCA*SCA+(BETA+DELTA)*(BETA-DELTA)*S1)-SCA)/S1        SHFAX     53
      CALL INTRPL(MC,PHI,C,1,PHO,COD)                                   SHFAX     54
      XOP=(ROC-BOD)/(COD-BOD)                                           SHFAX     55
      YOP=PHO/PHIO                                                      SHFAX     56
      XOFF=XOP*FLOAT(NA)+1.                                             SHFAX     57
      J=INT(XOFF)                                                       SHFAX     58
       XOFF=XOFF-FLOAT(J)                                               SHFAX     59
      YOFF=YOP*FLOAT(MA)+1.                                             SHFAX     60
      L=INT(YOFF)                                                       SHFAX     61
      YOFF=YOFF-FLOAT(L)                                                SHFAX     62
      IF(J.LT.NC) GO TO 25                                              SHFAX     63
      J=NA  $ XOFF=1.                                                   SHFAX     64
25    IF(J.GT.0) GO TO 30                                               SHFAX     65
       J=1  $ XOFF=0.                                                   SHFAX     66
30    IF(L.LT.MC) GO TO 35                                              SHFAX     67
      L=MA  $ YOFF=1.                                                   SHFAX     68
35    IF(L.GT.0) GO TO 40                                               SHFAX     69
      L=1  $ YOFF=0.                                                    SHFAX     70
40    XY=XOFF*YOFF                                                      SHFAX     71
      PN(N,M)=P(J,L)*(1.-XOFF-YOFF+XY)+P(J+1,L)*(XOFF-XY)+P(J,L+1)*(YOFF SHFAX     72
     1 -XY)+P(J+1,L+1)*XY                                               SHFAX     73
      DN(N,M)=D(J,L)*(1.-XOFF-YOFF+XY)+D(J+1,L)*(XOFF-XY)+D(J,L+1)*(YOFF SHFAX     74
     1 -XY)+D(J+1,L+1)*XY                                               SHFAX     75
      UNNM=U(J,L)*(1.-XOFF-YOFF+XY)+U(J+1,L)*(XOFF-XY)+U(J,L+1)*(YOFF   SHFAX1     9
     1 -XY)+U(J+1,L+1)*XY                                               SHFAX     77
      VNNM=V(J,L)*(1.-XOFF-YOFF+XY)+V(J+1,L)*(XOFF-XY)+V(J,L+1)*(YOFF   SHFAX1     10
     1 -XY)+V(J+1,L+1)*XY                                               SHFAX     79
      SCR=VNNM*SINPHO-UNNM*COSPHO                                       SHFAX1     11
      TCR=UNNM*SINPHO+VNNM*COSPHO                                       SHFAX1     12
      UN(N,M)=TCR*SINPHI(M)-SCR*COSPHI(M)                               SHFAX1     13
      VN(N,M)=SCR*SINPHI(M)+TCR*COSPHI(M)                               SHFAX1     14
20    CONTINUE                                                          SHFAX     80
      DO 100 M=1,MC                                                     SHFAX     81
      C(M)=CN(M)                                                        SHFAX     82
      DO 100 N=1,NC                                                     SHFAX     83
      P(N,M)=PN(N,M)                                                    SHFAX     84
      D(N,M)=DN(N,M)                                                    SHFAX     85
      U(N,M)=UN(N,M)                                                    SHFAX     86
      V(N,M)=VN(N,M)                                                    SHFAX     87
100   CONTINUE                                                          SHFAX     92
      DELTA=DELTA+SHIFDEL                                               SHFAX     93
      MY=MY-ZAS*FA $ MZ=MZ+ZAS*FY                                       SHFAX     94
      RETURN                                                            SHFAX     95
      END                                                               SHFAX     96
```

```
      SUBROUTINE SHFAXD(I,NDIM,MDIM,CV,CVP,CP,CZO,CON,CN,CNO,CPHIO)      SHFAXD     2
      DIMENSION CV(NDIM,MDIM,4),CVP(NDIM,MDIM,4)                         SHFAXD     3
      DIMENSION CZO(1),CON(1),CN(1),CNO(1),CPHIO(1),CP(1)                SHFAXD     4
      CALL SHFAX(I,NDIM,MDIM,CV(1,1,1),CV(1,1,2),CV(1,1,3),CV(1,1,4),    SHFAXD     5
     1   CVP(1,1,1),CVP(1,1,2),CP ,CZO,CON,CN,CNO,CPHIO)                 SHFAXD     6
      RETURN                                                             SHFAXD     7
      END                                                                SHFAXD     8
```

```
      SUBROUTINE BODY(M)                                                    BODY      2
C*********************************************************************** BODY      3
C                                                                           BODY      4
C     THIS ROUTINE COMPUTES THE RADIUS OF THE BODY AND DERIVATIVES ALONG  BODY      5
C     THE BODY AT A GIVEN  Z,PHI VALUE.  THE Z VALUE IS IN COMMON AND THE BODY      6
C     PHI VALUE IS PASSED AS AN ARGUMENT THROUGH THE PARAMETER M.         BODY      7
C       NOTE THAT Z IS ASSUMED TO BE INCREASING.                          BODY      8
C                                                                           BODY      9
C*********************************************************************** BODY     10
      COMMON /CBENT/ ZBB(25),ALNS,DST,AUQN,BETA,BSN,CENUF,DELII,DELTA       NEWCOM    5
     1   ,COSBN,EPSQ,ZMAXS,PID2,TANBN,IBN,MN,THETABN                       CBENT     3
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DFLZ                        CBODY     2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)             CBODY     3
      DIMENSION ZW(5),THETAW(5),ZL(5),THETAL(5),ZS(5),THETAS(5)           BODY     13
      DIMENSION ZCONE0(1),ZCONE(8),ACONE(8)                               BODY     14
      NAMELIST/BODYRD/NCONE,IRND,IEFL,ACONE,ZCONE,ZRND,RRND,ZFLARE,       BODY     15
     1 THETAFL,THETAGL,NW,NS,NL,IFW,IFS,IFL,ZW,ZS,ZL,THETAW,THETAS         BODY     16
     2  ,PHIS,THETAL,HFW,HFS,HFL,ZFW,ZFS,ZFL,FAW,FAS,FAL                  BODY     17
     3  ,IBN,THETABN,XLV,DELII,CENUF                                      CORR1     5
      DATA(ICONE=0)                                                       BODY     20
      DATA (ZLAST=0.)                                                     BODY     21
      IF (Z .NE. ZLAST) ICS=ICL=1                                         BODY     22
      PH=PHI(M)                                                           BODY     23
      IF (PH .LE. PI) GO TO 2                                             BODY     24
      PH=TPI-PH  $ SCC=-1.                                                BODY     25
      SINPHI(M)=-SINPHI(M)                                                BODY     26
      GO TO 3                                                             BODY     27
  2   SCC=1.                                                              BODY     28
C                                                                         BODY     29
C       COMPUTE THE RADIUS OF THE BASELINE BODY AT ANGLE PHI=PHI(M)       BODY     30
C                                                                         BODY     31
    3 IF (Z .GT. ZBAR) GO TO 29                                           BODY     32
      IF(Z.GT.ZBB(M)) GO TO 25                                            BODY     33
      BM=B(M)=SQRT(1.-(1.-Z)**2)                                          BODY     34
      BZ(M)=(1.-Z)/BM $BPHI(M)=BZPHI=BPHPHI=0.                            BODY     35
      BZZ=-1./BM**(1.+(1.-Z)*BZ(M)/BM)                                    BODY     36
      GO TO 1000                                                          BODY     37
C    ***   VERSION 3 OF TRANSITION REGION FOR BENT CONE    ***          BODY     38
   25 IF (Z .EQ. ZLAST) GO TO 42                                          BODY     39
      BETA=(COSBN+(Z-1.)*BSN)*BCN/AUQN                                    BODY     40
      DELTA=(SINBN*((1.-Z)*COSBN-BSN)+DELII*(COSBN**2-BSN**2))/AUQN       BODY     41
   42 IF (Z .LE. ZEN) GO TO 43                                            BODY     42
      IF (Z .EQ. ZLAST) GO TO 4                                           BODY     43
      BTMP=BBBAR+TANNS*Z                                                  BODY     44
      COSPHS=(DTIL2-Z*COTOV2)/BTMP                                        BODY     45
    4 IF (COSPHI(M) .LT. COSPHS) GO TO 43                                 BODY     46
      B(M)=BTMP                                                           BODY     47
      BZ(M)=TANNS                                                         BODY     48
      BZZ=BPHI(M)=BZPHI=BPHPHI=0.                                         BODY     49
      GO TO 1000                                                          BODY     50
   43 S2=SINPHI(M)**2                                                     BODY     51
      SCR=EPSQ*S2+1.                                                      BODY     52
      SCA=COSPHI(M)*DELTA                                                 BODY     53
      B(M)=BM=(SQRT(SCA*SCA+(BETA+DELTA)*(BETA-DELTA)*SCR)-SCA)/SCR       BODY     54
      TCR=SCR*BM+SCA                                                      BODY     55
      BZ(M)=BZM=(BETA*BETAP-DELTA*DELTAP-BM*(DELTAP*COSPHI(M)+BM*EPSQP*   BODY     56
     1 S2/2.))/TCR                                                        BODY     57
```

```
      BPHI(M)=BPHIM=BM*SINPHI(M)*(DELTA-BM*EPSQ*COSPHI(M))/TCR          BODY      58
      BZPHI=(BPHIM*COSPHI(M)*(DELTA*BZM/BM-DELTAP)-BM*SINPHI(M)*(EPSQ*  BODY      59
     1 BZM*COSPHI(M)-DELTAP*EPSQP*(BPHIM*SINPHI(M)+BM*COSPHI(M)))       BODY      60
     2  )/TCR                                                           BODY      61
      BPHPHI=(DELTA*(BPHIM**2*COSPHI(M)/BM+BM*COSPHI(M)+BPHIM*SINPHI(M) BODY      62
     1)+BM*EPSQ*(BM*(S2-COSPHI(M)**2)-3.*BPHIM*SINPHI(M)*COSPHI(M)))/   BODY      63
     2   TCR                                                            BODY      64
      BZZ=(-BZM*(2.*EPSQP*BM*S2+BZM*SCR+2.*DELTAP*COSPHI(M))+BETA*      BODY      65
     1  BETAPP+BETAP**2-DELTAP**2-DELTAPP*(DELTA+BM*COSPHI(M))-BM**2    BODY      66
     2 *EPSQPP*S2/2.)/TCR                                               BODY      67
      GO TO 1000                                                        BODY      68
   29 CONTINUE                                                          BODY      69
      IF(IRND.EQ.0) GO TO 30                                            BODY      70
      IF(Z.LT.ZRND) GO TO  30                                           BODY      71
      IRND=0                                                            BODY      72
      BRND=DELZ+ZRND*TANCO                                              BODY      73
      BBAR=COS(ACONE(NCONE)*PI/180.)                                    BODY      74
      RCEN=BRND-RRND*BBAR                                               BODY      75
      ZCEN=ZRND+RRND*SINCO                                              BODY      76
      ZBASE=ZCEN+RRND                                                   BODY      77
   30 IF(Z.LT.ZFLARE) GO TO 35                                          BODY      78
      IF(IEFL.EQ.0) GO TO 33                                            BODY      79
      IEFL=0                                                            BODY      80
      RC=DELZ+TANCO*ZFLARE                                              BODY      81
      TNFLR=TAN(THETAFL*RAD)/RC                                         BODY      82
      TNGLR=TAN(THETAGL*RAD)/RC                                         BODY      83
      RC2=RC*RC                                                         BODY      84
            GZSQ=TNGLR**2  $ FZSQ=TNFLR**2                              BODY      85
    33 IF (Z .EQ. ZLAST) GO TO 35                                       BODY      86
      GLZ=1.+(Z-ZFLARE)*TNGLR                                           BODY      87
      GSQ=GLZ*GLZ  $  GGZ=GLZ*TNGLR                                     BODY      88
      FLZ= 1.+(Z-ZFLARE)*TNFLR                                          BODY      89
      FSQ=FLZ*FLZ  $FFZ=FLZ*TNFLR                                       BODY      90
   35 IF(Z-ZCONE(ICONE))50,40,40                                        BODY      91
    40 BCONE=DELZ+TANCO*ZCONE(ICONE)                                    BODY      92
      ICONE=ICONE+1                                                     BODY      93
      TANCO=TAN(ACONE(ICONE)*PI/180.)                                   BODY      94
      DELZ=BCONE-TANCO*ZCONE(ICONE-1)                                   BODY      95
      IF(ZCONE(ICONE).LE.ZWW) GO TO 31                                  BODY      96
      ZWU=ZWW                                                           BODY      97
      BWU=DELZ+ZWU*TANCO                                                BODY      98
      TANW=TAN(THETAW(1)*RAD)                                           BODY      99
      ZWW=1.E08                                                         BODY     100
   31 IF(ZCONE(ICONE).LE.ZLL  ) GO TO 32                                BODY     101
      ZLU=ZLL                                                           BODY     102
      BLU=DELZ+ZLU*TANCO                                                BODY     103
      TANL=TAN(THETAL(1)*RAD)                                           BODY     104
      ZLL=1.E08                                                         BODY     105
   32 IF(ZCONE(ICONE).LE.ZSS) GO TO 35                                  BODY     106
      ZSU=ZSS                                                           BODY     107
      ZSS=1.E08                                                         BODY     108
      BSU=DELZ+ZSU*TANCO                                                BODY     109
      TANS=TAN(THETAS(1)*RAD)                                           BODY     110
      GO TO 35                                                          BODY     111
   50 IF(Z.LT.ZRND)26,27                                                BODY     112
   27 RR=RCEN+SQRT(RRND**2-(Z-ZCEN)**2)                                 BODY     113
      GO TO 100                                                         BODY     114
```

```
   26  IF(Z.LT.ZFLARE) GO TO 28                                          BODY      115
       COSSQ=COSPHI(M)**2  $ SINSQ=1.-COSSQ                              BODY      116
       COS2P=COSSQ-SINSQ  $ SIN2P=2.*SINPHI(M)*COSPHI(M)                 BODY      117
       RB=RC*SQRT(FSQ*COSSQ+GSQ*SINSQ)                                   BODY      118
       GO TO 100                                                         BODY      119
   28  RB=DELZ+Z*TANCO                                                   BODY      120
  100  IPTR=1                                                            BODY      121
C                                                                        BODY      122
C    COMPUTE THE DISTANCE TO THE SIDE-CUT PLANE                          BODY      123
C                                                                        BODY      124
       IF(IS-1) 140,110,120                                              BODY      125
  110  IF(Z.LT.ZS(IS))GO TO 140                                          BODY      126
       IS=IS+1                                                           BODY      127
  120  IF(Z.LT.ZS(IS))GO TO 130                                          BODY      128
       BSU=BSU+(ZS(IS)-ZSU)*TANS                                         BODY      129
       ZSU=ZS(IS)                                                        BODY      130
       TANS=TAN(THETAS(IS)*RAD)  $IS=IS+1                                BODY      131
       IF(IS.NE.3) GO TO 120                                             BODY      132
       IF(IFS.EQ.0) GO TO 120                                            BODY      133
       BFSI=BSU $ BFSF=BSU+TANFS*ZFS $ ZFS=ZFS+ZSU                       BODY      134
       ZFSI=ZSU                                                          BODY      135
       GO TO 120                                                         BODY      136
  130  IF(ABS(PH-PHIS)-PID2.GT.-1.E-05) GO TO 140                        BODY      137
       RS=(BSU+(Z-ZSU)*TANS)/COS(PHIS-PH)                                BODY      138
       IF(RS.GE.RB)GO TO 140                                             BODY      139
       IPTR=2  $ RB=RS                                                   BODY      140
C                                                                        BODY      141
C    COMPUTE THE DISTANCE TO THE LEE-CUT PLANE                           BODY      142
C                                                                        BODY      143
  140  IF(IL-1)180,150,160                                               BODY      144
  150  IF(Z.LT.ZL(1))GO TO 180                                           BODY      145
       IL=IL+1                                                           BODY      146
  160  IF(Z.LT.ZL(IL)) GO TO 170                                         BODY      147
       BLU=BLU+(ZL(IL)-ZLU)*TANL  $ZLU=ZL(IL)  $TANL=TAN(THETAL(IL)*RAD) BODY      148
       IL=IL+1                                                           BODY      149
       IF(IL.NE.3) GO TO 160                                             BODY      150
       IF(IFL.EQ.0) GO TO 160                                            BODY      151
       BFLI=BLU $ BFLF=BLU+TANFL*ZFL $ ZFL=ZFL+ZLU                       BODY      152
       ZFLI=ZLU                                                          BODY      153
       GO TO 160                                                         BODY      154
  170  IF(PH.LT.PID2+1.E-05) GO TO 180                                   BODY      155
       RL=-(BLU+(Z-ZLU)*TANL)/COSPHI(M)                                  BODY      156
       IF(RL.GE.RB) GOTO  180                                            BODY      157
       IPTR=3   $RB=RL                                                   BODY      158
C                                                                        BODY      159
C    COMPUTE THE DISTANCE TO THE WIND-CUT PLANE                          BODY      160
C                                                                        BODY      161
  180  IF(IW-1)300,190,200                                               BODY      162
  190  IF(Z.LT.ZW(1))GO TO 300                                           BODY      163
       IW=IW+1                                                           BODY      164
  200  IF(Z.LT.ZW(IW))GO TO 210                                          BODY      165
       BWU=BWU+(ZW(IW)-ZWU)*TANW                                         BODY      166
       ZWU=ZW(IW)  $TANW=TAN(THETAW(IW)*RAD)  $IW=IW+1                   BODY      167
       IF(IW.NE.3) GO TO 200                                             BODY      168
       IF(IFW.EQ.0) GO TO 200                                            BODY      169
       BFWI=BWU  $ BFWF=BWU+TANFW*ZFW  $ ZFW=ZFW+ZWU                     BODY      170
       ZFWI=ZWU                                                          BODY      171
```

```
      GO TO 200                                                          BODY      172
  210 IF(PH.GT.PID2-1.E-05) GOTO 300                                     BODY      173
      RW=(BWU+(Z-ZWU)*TANW)/COSPHI(M)                                    BODY      174
      IF(RW.GE.RB) GO TO 300                                             BODY      175
      IPTR=4  $RB=RW                                                     BODY      176
  300 B(M)=RB                                                            BODY      177
C                                                                        BODY      178
C     ITPR NOW CONTAINS A POINTER INDICATING WHICH OF THE ABOVE COMPUTED  BODY      179
C     DISTANCES IS MINIMAL.                                              BODY      180
C         IF IPTR=1     THEN THE MINIMAL DISTANCE IS TO THE BASELINE BODY. BODY    181
C         IF IPTR=2     THEN THE MINIMAL DISTANCE IS TO THE SIDE-CUT PLANE BODY    182
C         IF IPTR=3     THEN THE MINIMAL DISTANCE IS TO THE LEE-CUT PLANE  BODY    183
C         IF IPTR=4     THEN THE MINIMAL DISTANCE IS TO THE WIND-CUT PLANE BODY    184
C                                                                        BODY      185
C     THE ROUTINE THEN TRANSFERS TO THE APPROPRIATE SECTION OF THE CODE  BODY      186
C     FOR COMPUTATION OF DERIVATIVES.                                    BODY      187
C                                                                        BODY      188
      GO TO (900,400,500,600),IPTR                                       BODY      189
C                                                                        BODY      190
C          SIDE CUT                                                      BODY      191
C     CHECK FOR FLAP AND COMPUTE QUANTITIES NEEDED TO CALCULATE THE      BODY      192
C     DERIVATIVES.                                                       BODY      193
C                                                                        BODY      194
  400 IF(IFS*IS.LE.2) GO TO 460                                          BODY      195
      IF(ICS.EQ.0) GO TO 430                                             BODY      196
      ICS=0                                                              BODY      197
      RS=RS*COS(PH-PHIS)                                                 BODY      198
      PHIFS=ATAN(HFS/RS)                                                 BODY      199
      IF(Z.GT.ZFS)410,420                                                BODY      200
  410 RSF=BFSF                                                           BODY      201
      PHITS=ATAN(HFS/RSF)                                                BODY      202
      TANFS=0                                                            BODY      203
       GO TO 425                                                         BODY      204
  420 RSF=BFSI+(Z-ZFSI)*TANFS                                            BODY      205
      PHITS=ATAN(HFS/RSF)                                                BODY      206
  425 IF(RSF.GT.RS) GO TO 430                                            BODY      207
      IFS=0  $ GO TO 460                                                 BODY      208
  430 IF(ABS(PH-PHIS).GE.PHIFS) GO TO 460                                BODY      209
      IF(ABS(PH-PHIS).GT.PHITS) GO TO 450                                BODY      210
      COSU=COS(PH-PHIS)                                                  BODY      211
      TANU=TAN(PH-PHIS)                                                  BODY      212
      B(M)=RB=RSF/COSU                                                   BODY      213
      BZ(M)=TANFS/COSU                                                   BODY      214
      GO TO 800                                                          BODY      215
  450 SINU=SIN(PH-PHIS)   $COSU=COS(PH-PHIS)                             BODY      216
      B(M)=BM=ABS(HFS/SINU)                                              BODY      217
      GO TO 700                                                          BODY      218
  460 COSU=COS(PH-PHIS)                                                  BODY      219
      TANU=SIN(PH-PHIS)/COSU                                             BODY      220
      BZ(M)=TANS/COSU                                                    BODY      221
      GO TO 800                                                          BODY      222
C                                                                        BODY      223
C          LEE CUT                                                       BODY      224
C     CHECK FOR FLAP AND COMPUTE QUANTITIES NEEDED TO CALCULATE THE      BODY      225
C     DERIVATIVES.                                                       BODY      226
C                                                                        BODY      227
  500 IF(IFL*IL.LE.2) GO TO 570                                          BODY      228
```

```
      IF(ICL.EQ.0) GO TO 540                                            BODY      229
      ICL=0                                                             BODY      230
      RL=-RL*COSPHI(M)                                                  BODY      231
      PHIFL=ATAN(HFL/RL)                                                BODY      232
      IF(Z.LE.ZFL) GO TO 530                                            BODY      233
      RLF=BFLF                                                          BODY      234
      PHITL=ATAN(HFL/RLF)                                               BODY      235
      TANFL=0                                                           BODY      236
      GO TO 535                                                         BODY      237
  530 RLF=BFLI+(Z-ZFLI)*TANFL                                           BODY      238
      PHITL=ATAN(HFL/RLF)                                               BODY      239
  535 IF(RLF.GT.RL) GO TO 540                                           BODY      240
      IFL=0                                                             BODY      241
      GO TO 570                                                         BODY      242
  540 IF(ABS(PI-PH).GE.PHIFL) GO TO 570                                 BODY      243
      IF(ABS(PI-PH).GT.PHITL) GO TO 560                                 BODY      244
      COSU=COSPHI(M)                                                    BODY      245
      TANU=SINPHI(M)/COSU                                               BODY      246
      B(M)=RB=-RLF/COSU                                                 BODY      247
      BZ(M)=-TANFL/COSU                                                 BODY      248
      GO TO 800                                                         BODY      249
  560 SINU=SINPHI(M)                                                    BODY      250
      COSU=COSPHI(M)                                                    BODY      251
      B(M)=BM=HFL/SINU                                                  BODY      252
      GO TO 700                                                         BODY      253
  570 BZ(M)=-TANL/COSPHI(M)                                             BODY      254
      TANU=SINPHI(M)/COSPHI(M)                                          BODY      255
      GO TO 800                                                         BODY      256
C                                                                       BODY      257
C           WIND CUT                                                    BODY      258
C     CHECK FOR FLAP AND COMPUTE QUANTITIES NEEDED TO CALCULATE THE     BODY      259
C     DERIVATIVES.                                                      BODY      260
C                                                                       BODY      261
  600 IF(IFW*IW.LE.2) GO TO 710                                         BODY      262
      IF (Z .EQ. ZLAST) GO TO 630                                       BODY      263
      PHIFW=ATAN(HFW/RW)                                                BODY      264
      IF(Z.LE.ZFW) GO TO 620                                            BODY      265
      RWF=BFWF                                                          BODY      266
      PHITW=ATAN(HFW/RWF)                                               BODY      267
      TANFW=0                                                           BODY      268
      GO TO 625                                                         BODY      269
  620 RWF=BFWI+(Z-ZFWI)*TANFW                                           BODY      270
      PHITW=ATAN(HFW/RWF)                                               BODY      271
  625 IF(RWF.GT.RB) GO TO 630                                           BODY      272
      IFW=0                                                             BODY      273
      GO TO 710                                                         BODY      274
  630 IF(ABS(PH).GE.PHIFW) GO TO 710                                    BODY      275
      IF(ABS(PH).GT.PHITW) GO TO 650                                    BODY      276
      COSU=COSPHI(M)                                                    BODY      277
      TANU=SINPHI(M)/COSU                                               BODY      278
      BZ(M)= TANFW/COSU                                                 BODY      279
      B(M)=RB=RWF/COSPHI(M)                                             BODY      280
      GO TO 800                                                         BODY      281
  650 SINU=SINPHI(M)                                                    BODY      282
      COSU=COSPHI(M)                                                    BODY      283
      B(M)=BM=HFW/SINU                                                  BODY      284
C                                                                       BODY      285
```

```
C     COMPUTE DERIVATIVES ON THE SIDE OF THE FALP.                          BODY      286
C                                                                           BODY      287
  700 BZ(M)=BZZ=BZPHI=0.                                                    BODY      288
      BPHI(M)=-BM*COSU/SINU                                                 BODY      289
      BPHPHI=BM*(2./(SINU*SINU)-1.)                                         BODY      290
      GO TO 1000                                                            BODY      291
  710 BZ(M)=TANW/COSPHI(M)                                                  BODY      292
      TANU=SINPHI(M)/COSPHI(M)                                              BODY      293
C                                                                           BODY      294
C     COMPUTE THE DERIVATIVES ON CUT OR TOP OF FLAP.                        BODY      295
C                                                                           BODY      296
  800 BPHI(M)=TANU*RB                                                       BODY      297
      BZZ=0.                                                                BODY      298
      BZPHI=BZ(M)*TANU                                                      BODY      299
      BPHPHI=RB*2.*BPHI(M)*TANU                                             BODY      300
      GO TO 1000                                                            BODY      301
C                                                                           BODY      302
C     COMPUTE THE DERIVATIVES ON BASELINE BODY.                             BODY      303
C                                                                           BODY      304
  900 IF(Z.LE.ZRND) GO TO 905                                               BODY      305
      RZ=RB-RCEN                                                            BODY      306
      BZM=BZ(M)=(ZCEN-Z)/RZ                                                 BODY      307
      BZZ=-(1.+BZM*BZM)/RZ                                                  BODY      308
      BZPHI=BPHPHI=BPHI(M)=0.                                               BODY      309
      RETURN                                                                BODY      310
  905 IF(Z.LT.ZFLARE) GO TO 910                                             BODY      311
      BZM=BZ(M)=RC2*(FFZ*COSSQ+GGZ*SINSQ)/RB                                BODY      312
      BZZ=(RC2*(FZSQ*COSSQ+GZSQ*SINSQ)-BZM*BZM)/RB                          BODY      313
      BP=BPHI(M)=.5*RC2*SIN2P*(GSQ-FSQ)/RB                                  BODY      314
      BPHPHI=(RC2*(GSQ-FSQ)*COS2P-BP*BP)/RB                                 BODY      315
      BZPHI=(RC2*SIN2P*(GGZ-FFZ)-BP*BZM)/RB                                 BODY      316
      GO TO 1000                                                            BODY      317
  910 BZ(M)=TANCO  $ BZPHI=BPHPHI=BZZ=BPHI(M)=0.                            BODY      318
 1000 SINPHI(M)=SCC*SINPHI(M)                                               BODY      319
      BPHI(M)=SCC*BPHI(M)  $ BZPHI=SCC*BZPHI                                BODY      320
      ZLAST=Z                                                               BODY      321
      RETURN                                                                BODY      322
      ENTRY BODYR                                                           BODY      323
C*************************************************************************BODY      324
C                                                                           BODY      325
C       THIS PART OF THE ROUTINE READS IN DATA DEFINING THE BODY            BODY      326
C       AND COMPUTES VARIOUS PARAMETERS.                                    BODY      327
C                                                                           BODY      328
C*************************************************************************BODY      329
C                                                                           BODY      330
C         NCONE DETERMINES THE NUMBER OF CONIC SECTIONS                     BODY      331
C         IRND DETERMINES WHETHER THE END OF THE BODY IS ROUNDED.           BODY      332
C             (IRND=1 IF ROUNDED AND 0 TOHERWISE)                           BODY      333
C       IEFL DETERMINES WHETHER THERE IS AN ELLIPTIC FLARE ON THE           BODY      334
C             END OF THE BODY.  (IEFL=1 IF THERE IS AN ELLIPTIC FLARE       BODY      335
C             AND IS 0 OTHERWISE.)                                          BODY      336
C     NEXT, FOR EACH CONIC SECTION, INPUT THE ANGLE OF THE CONE AND THE     BODY      337
C     Z VALUE WHERE IT ENDS.                                                BODY      338
C     IF THE BODY IS ROUNDED, INPUT THE Z LOCATION OF THE BEGINNING OF      BODY      339
C     THE ROUND AND THE RADIUS OF THE ROUNDING.                             BODY      340
C     IF THE AFTERBODY IS FLARED, INPUT THE Z LOCATION OF THE BEGINNING O   BODY      341
C     THE FLARE AND THE ANGLE OF EXPANSION OF THE PHI=0 AND 180 DEGREE      BODY      342
```

```
C     AXIS AND THE ANGLE OF EXPANSION FO THE PHI=90 DEGREE AXIS.          BODY      343
C                                                                         BODY      344
C         NW IS THE NUMBER OF SECTION OF THE WIND CUT                     BODY      345
C         NS IS THE NUMBER OF SECTION OF THE SIDE CUT                     BODY      346
C         NL IS THE NUMBER OF SECTION OF THE LEE CUR                      BODY      347
C         NL IS THE NUMBER OF SECTION OF THE LEE CUT                      BODY      348
C         IFW IS ONE IF THERE IS A WIND FLAP 0 OTHERWISE                  BODY      349
C         IFS IS ONE IF THERE IS A SIDE FLAP 0 OTHERWISE                  BODY      350
C         IFL IS ONE IF THERE IS A LEE FLAP 0 OTHERWISE                   BODY      351
C                                                                         BODY      352
C     NOTE THAT ALL FLAPS BEGIN ON THE SECOND PART OF THE CORRESPONDING   BODY      353
C     CUT.                                                                BODY      354
C                                                                         BODY      355
C     NEXT, FOR EACH CUT SECTION INPUT THE Z LOCATION OF THE BEGINNING    BODY      356
C     OF THAT SECTION AND THE ANGLE OF THAT SECTION                       BODY      357
C                                                                         BODY      358
C     FINALLY, FOR EACH EXISTING FLAP INPUT THE HALF-WIDTH OF THE FLAP,   BODY      359
C     THE LENGTH OF THE FLAP ALONG THE Z AXIS, AND THE FLAP ANGLE.        BODY      360
C                                                                         BODY      361
      PI=4.*ATAN(1.) $ RAD=PI/180.                                        BODY      362
      NCONE=1  $ ZRND=ZFLARE=1.E08                                        BODY      363
      NW=NS=NL=IFW=IFS=IFL=IEFL=IRND=0$IW=IL=IS=0                         BODY      364
      CENUF=.5                                                            BODY      365
      HN=0.  $ IBN=0                                                      BODY      366
      DELII=0.                                                            BODY      367
      PID2=PI/2.                                                          BODY      368
      PHIS=90.                                                            BODY      369
      FAW=FAS=FAL=0.                                                      BODY      370
      READ(5,BODYRD)                                                      BODY      371
      PID2=PI/2.                                                          BODY      372
      TPI=PI+PI                                                           BODY      373
      ZCONE0(1)=0.                                                        BODY      374
      ZCONE(NCONE)=1.E08                                                  BODY      375
 1030 CONE=ACONE(1)                                                       BODY      376
      ALNS=CONE                                                           COPR1       6
      SINCO=SIN(CONE*RAD)  $ COSCO=COS(CONE*RAD)                          BODY      377
      IF(IBN.EQ.1) GO TO 1040                                             BODY      378
 1045 CONTINUE                                                            BODY      379
      ZBAR=1.-SINCO  $ BBAR=COSCO  $ GO TO 1050                           BODY      380
C    ***   COMPUTE PARAMETERS FOR BENT NOSE   ***                         BODY      381
 1040 ALNS=ALNS*RAD  $ THETABN=THETABN*RAD                                BODY      382
      BCN=COS(ALNS)  $ BSN=SIN(ALNS)                                      BODY      383
      COSBN=COS(THETABN)  $ SINBN=SIN(THETABN)                            BODY      384
      TANBN=SINBN/COSBN                                                   BODY      385
      TANNS=BSN/BCN $ SNO2=SIN(THETABN/2.) $ CNO2=COS(THETABN/2.)         BODY      386
      DUM1=CNO2**2-BSN**2 $ DUM2=XLV*CNO2/DUM1                            BODY      387
      DUM3=1.-COSBN/BSN $ DUM4=CNO2*(COSBN-BSN**2) $ DUM5=BCN*BSN*SNO2    BODY      388
      ZEN=DUM3+DUM2*(DUM4-DUM5) $ ZBAR=DUM3+DUM2*(DUM4+DUM5)              BODY      389
      DUM6=(DUM2*(CNO2*BCN**2-DUM5))*TANNS $ BBBAR=DUM6-ZEN*TANNS         BODY      390
      BBAR=BBBAR+ZBAR*TANNS $ COTOV2=CNO2/SNO2                            BODY      391
      DTIL2=DUM6+ZEN*COTOV2 $ HN=SINBN*(DUM2*CNO2-1./BSN)-DELII           BODY      392
      AUQN=(COSBN*BCN)**2-(BSN*SINBN)**2                                  BODY      393
      EPSQ=SINBN**2/AUQN  $ EPSQP=EPSQPP=0.                               BODY      394
      BETAP=BCN*BSN/AUQN                                                  BODY      395
      BETAPP=0.  $ DELTAP=-SINBN*COSBN/AUQN                               BODY      396
      DELTAPP=0.                                                          BODY      397
      DST=ZEN-.5 $ ZMAXS=CENUF*TANBN                                      BODY      398
```

```
 1050 CONTINUE                                                          BODY      399
      TANCO=SINCO/COSCO                                                 BODY      400
      DEL7=BBAR-TANCO*ZBAR                                              BODY      401
      ZL(NL+1)=1.E08                                                    BODY      402
      ZS(NS+1)=1.E08                                                    BODY      403
      ZW(NW+1)=1.E08                                                    BODY      404
      IF(NW.NE.0) IW=1                                                  BODY      405
      IF(NS.NE.0) IS=1                                                  BODY      406
      IF(NL.NE.0) IL=1                                                  BODY      407
      PHIS=PHIS*RAD                                                     BODY      408
      TANFW=TAN(FAW*RAD)                                                BODY      409
      TANFS=TAN(FAS*RAD)                                                BODY      410
      TANFL=TAN(FAL*RAD)                                                BODY      411
 1640 ZLL=ZL(1) $ ZWW=ZW(1)  $ ZSS=ZS(1)                                BODY      412
      RETURN                                                            BODY      413
 2000 FORMAT(I5)                                                        BODY      414
 2001 FORMAT(2F10.4)                                                    BODY      415
 2002 FORMAT(3F10.4)                                                    BODY      416
      ENTRY BODYW                                                       BODY      417
C*********************************************************************  BODY      418
C                                                                       BODY      419
C     THIS PART OF THE ROUTINE PRINTS OUT INFORMATION ABOUT THE BODY.   BODY      420
C                                                                       BODY      421
C*********************************************************************  BODY      422
      IVERSON=3                                                         BODY      423
      WRITE (6,3000) IVERSON                                            BODY      424
      IF(IBN.EQ.1) GO TO 3020                                           BODY      425
      WRITE(6,3005)ZBAR,BBAR                                            BODY      426
      GO TO 3021                                                        BODY      427
 3020 WRITE(6,3100)ALNS/RAD,THETABN/RAD,ZEN                             BODY      428
 3100 FORMAT(11X,*FORE BODY IS A SPHERICALLY BLUNTED CONE OF *,F10.4,    BODY      429
     1  * DEGREES BENT AT AN ANGLE OF *,F10.4/11X,*THE BENT CONE ENDS AT BODY      430
     2 *,F15.7)                                                         BODY      431
      WRITE(6,3110)ZBAR,BBAR                                            BODY      432
 3110 FORMAT(11X,*THE BODY IS SMOOTHED TO THE AFT BODY BEGINNING AT*,    BODY      433
     1  E15.7,* AND HAVING RADIUS *,E15.7/11X,* THE AFT BODY IS A MULTIP BODY      434
     2LE CONIC WITH *)                                                  BODY      435
 3021 CONTINUE                                                          BODY      436
      WRITE(6,3010)(ACONE(I),ZCONE(I),I=1,NCONE)                        BODY      437
      IF(IRND.EQ.1)  WRITE(6,3025) RRND,ZRND                            BODY      438
      IF(IEFL.EQ.1) WRITE(6,3030) ZFLARE,THETAFL,THETAGL                BODY      439
      IF(NW.EQ.0) GO TO 3060                                            BODY      440
      WRITE(6,3035)                                                     BODY      441
      WRITE(6,3040)(THETAW(I),ZW(I),I=1,NW)                             BODY      442
      IF(IFW.EQ.1) WRITE(6,3050)HFW,ZFW,FAW                             BODY      443
 3060 IF(NS.EQ.0) GO TO 3090                                            BODY      444
      PHIS=PHIS/RAD                                                     BODY      445
      WRITE(6,3070) PHIS                                                BODY      446
      PHIS=PHIS*RAD                                                     BODY      447
      WRITE(6,3040)(THETAS(I),ZS(I),I=1,NS)                             BODY      448
      IF(IFS.EQ.1) WRITE(6,3050) HFS,ZFS,FAS                            BODY      449
 3090 IF(NL.EQ.0) RETURN                                                BODY      450
      WRITE(6,3095)                                                     BODY      451
      WRITE(6,3040)(THETAL(I),ZL(I),I=1,NL)                             BODY      452
      IF(IFL.EQ.1) WRITE(6 ,3050) HFL,ZFL,FAL                           BODY      453
      RETURN                                                            BODY      454
 3000 FORMAT(1H0,20X,*PROGRAM BODY*,6X,*VERSION*,I4)                    BODY      455
```

```
3005 FORMAT(11X,*BODY IS SPHERICALLY BLUNTED AND SPHERE ENDS AT Z=*,     BODY      456
    1 E15.7,2X,* WITH B=*,E15.7/11X,* AFT BODY IS A MULTIPLE CONIC WITH BODY      457
    2 *)                                                                BODY      458
3010 FORMAT(16X,*ANGLE*,F10.4,* UP TO *,F10.4)                          BODY      459
3025 FORMAT(11X,*THE REAR OF THE BODY IS ROUNDED WITH RADIUS*,F10.4/     BODY      460
    1  11X,* THE ROUNDING BEGINS AT*, F10.4)                            BODY      461
3030 FORMAT(11X,*THERE IS AN ELLIPTIC FLARE BEGINNING AT*, F10.4/       BODY      462
    1 11X,*THE WIND-LEE AXIS EXPANDS WITH ANGLE*, F10.4,*  AND THE SIDE BODY      463
    2  AXIS EXPANDS WITH ANGLE*, F10.4          )                       BODY      464
3035 FORMAT(11X,*THERE IS A WIND CUT OF*)                               BODY      465
3040 FORMAT(16X,*ANGLE*,F10.4,* BEGINNING AT*,F10.4)                    BODY      466
3050 FORMAT(20X,*WITH A FLAP OF HALF-WIDTH*,F10.4,* LENGTH ALONG Z-AXIS BODY      467
    1  *,F10.4,*  AT*,F10.4,* DEGREES*)                                 BODY      468
3070 FORMAT(11X,*THERE IS A SIDE CUT CENTERED AT LONGITUDE*,F10.4, * OF BODY      469
    1 *)                                                                BODY      470
3095 FORMAT(11X,* THERE IS A LEE CUT OF*)                               BODY      471
     END                                                                BODY      472
```

```
      SUBROUTINE FIELD                                                  FIELD      2
C                                                                       FIELD      3
C      FIELD PRINTS THE DATA AT SOME FIX Z                              FIELD      4
C                                                                       FIELD      5
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                        NEWCOM     1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)               NEWCOM     2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)    NEWCOM     3
      COMMON CU(4,20,25),CUP(4,20,25)                                   NEWCOM     4
C   ***   END OF BLANK COMMON   ***                                     CD3CSS    32
      COMMON /CBENT/ ZBB(25),ALNS,DST,AUON,BFTA,BSN,CENUF,DELII,DELTA   NEWCOM     5
     1   ,COSBN,EPSQ,ZMAXS,PID2,TANBN,IBN,HN,THETABN                    CBENT      3
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                      CBODY      2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)           CBODY      3
      COMMON /COUT/ ACH,ATTA,YAW,ZEND,XINDEF,VINF,SINF                  COUT       2
     1   ,NTARGET,TARGETZ(100)                                          COUT       3
C                                                                       FIELD      8
      COMMON/RGASS/AX,HX,TX,RRX,GX,NTEST,NGAS,NFIRST                    FIELD      9
      WRITE (6,3000) ACH,ATTA,YAW                                       FIELD     10
      DO 100 M=1,MC                                                     FIELD     11
      H=PHI(M)/RAD $ WRITE (6,3010) M,H                                 FIELD     12
      WRITE(6,3800) HN                                                  FIELD     13
      WRITE (6,3600) K,Z,B(M),BZ(M),BPHI(M),C(M),CZ(M),CPHI(M)          FIELD     14
      WRITE (6,3700)                                                    FIELD     15
      DO 50 N=1,NC                                                      FIELD     16
      L=NC-N+1                                                          FIELD     17
      IF ((P(L,M) .GT. 0.) .AND. (D(L,M) .GT. 0.)) GO TO 10             FIELD     18
      AMACH=SX=XINDEF                                                   FIELD     19
      GO TO 25                                                          FIELD     20
   10 CONTINUE                                                          FIELD     21
      CALL RGAS(P(L,M),D(L,M),SX,4)                                     FIELD     22
      AMACH=SQRT(U(L,M)**2+V(L,M)**2+W(L,M)**2)/AX                      FIELD     23
   25 CONTINUE                                                          FIELD     24
      GAMC=1./(1.-P(L,M)/(HX*D(L,M)))                                   FIELD     25
      WRITE (6,3400) R(L,M),W(L,M),U(L,M),V(L,M),P(L,M),D(L,M),         FIELD     26
     1SX,AMACH,GAMC                                                     FIELD     27
   50 CONTINUE                                                          FIELD     28
  100 CONTINUE                                                          FIELD     29
      RETURN                                                            FIELD     30
 3000 FORMAT(1H1,*MACH NO IS*,1PE15.7,5X,                               FIELD     31
     1   *ANGLE OF ATTACK IS*,1PE15.7,5X,*ANGLE OF SIDESLIP IS*,1PE15.7) FIELD     32
 3010 FORMAT(1H0,*PLANE*,I4,3X,*ANGLE IS*,F7.2,* DEGREES*)              FIELD     33
 3100 FORMAT(1H+,50X,*WINDWARD PLANE*)                                  FIELD     34
 3200 FORMAT(1H+,50X,*LEEWARD PLANE*)                                   FIELD     35
 3400 FORMAT(1H ,1P9E14.4)                                              FIELD     36
 3600 FORMAT(1H0,*STATION*,I5,4X,*Z IS*,1PE15.7,4X,*R IS*,1PE15.7,4X,   FIELD     37
     1          *BZ IS*,1PE15.7,4X,*BPHI IS*,1PE15.7,/,                FIELD     38
     2       17X,*C IS*,1PE15.7,4X,*CZ IS*,1PE15.7,4X,*CPHI IS*,1PE15.7) FIELD     39
 3700 FORMAT(1H0,8X,1HR,13X,1HW,13X,1HU,13X,1HV,13X,1HP,12X,3HRHO,      FIELD     40
     1   13X,1HS,13X,1HM,11X,5HGAMMA)                                   FIELD     41
 3800 FORMAT(11X,*THE AXIS IS SHIFTED UP*,F10.4,* UNITS*)               FIELD     42
      END                                                               FIELD     43
```

```
      SUBROUTINE OUT                                                    OUT       2
C                                                                       OUT       3
C      OUT OUTPUTS THE WALL PRESSURES AND THE FORCES AND MOMENTS.       OUT       4
C      THIS ROUTINE IS EXECUTED AFTER CALCULATIONS ARE COMPLETED.       OUT       5
C                                                                       OUT       6
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                        NEWCOM    1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)               NEWCOM    2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)    NEWCOM    3
      COMMON CU(4,20,25),CUP(4,20,25)                                   NEWCOM    4
C   ***   END OF BLANK COMMON   ***                                     CD3CSS   32
      COMMON /COUT/ ACH,ATTA,YAW,ZEND,XINDEF,VINF,SINF                  COUT      2
     1   ,NTARGET,TARGETZ(100)                                         COUT      3
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                      CBODY     2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)          CBODY     3
      COMMON /CINTEG/ FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ        CINTEG    2
     1   ,DYD3,MA,GY(25)                                               CINTEG    3
      REAL MX,MY,MZ,MXZ,MYZ,MZZ                                         CINTEG    4
C                                                                       OUT       8
      COMMON/RGASS/AX,HX,TX,RRX,GX,NTEST,NGAS                           OUT       9
      DIMENSION ZS(2),FNS(2),FYS(2),FAS(2),MXS(2),MYS(2),MZS(2)         OUT      10
      REAL MXS,MYS,MZS                                                  OUT      11
      DIMENSION PB(1)                                                   OUT      12
      EQUIVALENCE (PB,B)                                                OUT      13
      NAMELIST /OUTRD/ ZREF,AREF,ZC,Z0,IPCID                            OUT      14
      IF (ZREF .EQ. 0.) ZREF=ZEND+Z0                                    CORR1     7
      IF (AREF .GT. 0.) GO TO 10                                        OUT      16
      Z=ZEND $ CALL BODY(1) $ RB=DELZ+TANCO*ZEND $ AREF=PI*RB**2        CORR1     8
   10 REWIND 16 $ NPTS=0 $ M1=1 $ MCMX=MC $ M1P14=M1+14                 OUT      18
      CONT1=.5*DINF*VINF**2/PINF                                        OUT      19
   25 NPTS=NPTS+1                                                       OUT      20
   30 MCS=MC                                                            OUT      21
      READ (16) NC,MC,ATTA,YAW,ACH,GAMMA,PINF,DINF,PHIO,K,Z             OUT      22
     A   ,NGAS,NTEST,RRX                                                OUT      23
     1   ,FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ                    OUT      24
     2    ,(PHI(M),DUM,DUM,DUM,M=1,MC)                                 OUT      25
     3    ,(DUM,DUM,DUM,DUM,PB(M),DUM,M=1,MC)                          OUT      26
      IF (EOF(16)) 200,50                                               OUT      27
   50 M2=MIN0(MC,M1P14) $ IF (M2 .LT. M1) GO TO 30                      OUT      28
      IF (MC .NE. MCS) NPTS=1                                           OUT      29
      IF (MOD(NPTS-1,38) .NE. 0) GO TO 75                               AERO      1
      WRITE (6,3000) ACH,ATTA,YAW,Z0                                    OUT      31
 3000 FORMAT(1H1,10X,*MACH NO =*,F8.3,5X                                OUT      32
     1   ,*ANGLE OF ATTACK =*,F8.3,5X,*ANGLE OF SIDESLIP =*,F8.3        OUT      33
     2   ,5X,*Z0 =*,F8.3)                                               OUT      34
      IF (IPCID .EQ. 0) WRITE (6,3020)                                  OUT      35
 3020 FORMAT(1H0,35X,*S U R F A C E   P R E S S U R E   R A T I O*)     OUT      36
      IF (IPCID .NE. 0) WRITE (6,3025)                                  OUT      37
 3025 FORMAT(1H0,30X,*S U R F A C E   P R E S S U R E   *,              OUT      38
     1   *C O E F F I C I E N T*)                                       OUT      39
      DO 60 M=M1,M2                                                     OUT      40
   60 PHI(M)=PHI(M)/RAD                                                 OUT      41
      WRITE (6,3030) (PHI(M),M=M1,M2)                                   OUT      42
 3030 FORMAT(1H ,5X,4HZ+Z0,15F8.1)                                      OUT      43
      WRITE (6,3040)                                                    OUT      44
 3040 FORMAT(1H )                                                       OUT      45
   75 DO 62 M=M1,M2                                                     OUT      46
   62 PB(M)=PB(M)/PINF                                                  OUT      47
```

```
      MZ=MZS(I1)+(MZS(I2)-MZS(I1))*CONT1                                OUT       105
      CN=XK0*FN $ CA=XK0*FA $ CY=XK0*FY                                 OUT       106
      CMX=XK1*(MX+ZC*FY) $ CMY=XK1*(MY-ZC*FN) $ CMZ=XK1*MZ              OUT       107
      XCPP=XINDEF $ XCPY=XINDEF                                         OUT       108
      IF (CN .NE. 0.) XCPP=ZC/ZREF+CMY/CN+Z0/ZREF                       OUT       109
      IF (CY .NE. 0.) XCPY=ZC/ZREF-CMX/CY+Z0/ZREF                       OUT       110
      CY=-CY $ CMX=-CMX $ CMY=-CMY $ CMZ=-CMZ                           OUT       111
      ZZ=ZT+Z0                                                          OUT       112
      WRITE (6,3330) ZZ,CN,CA,CY,CMX,CMY,CMZ,XCPP,XCPY                  OUT       113
      GO TO 300                                                         OUT       114
  325 NPTS=0 $ REWIND 18                                                OUT       115
  350 NPTS=NPTS+1                                                       OUT       116
      READ (18) Z,FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ            OUT       117
      IF (EOF(18)) 425,375                                              OUT       118
  375 IF (MOD(NPTS-1,38) .NE. 0) GO TO 400                              AERO        2
      WRITE (6,3200)                                                    OUT       120
      WRITE (6,3340)                                                    OUT       121
      WRITE (6,3350)                                                    OUT       122
  400 CNZ=XK0*FNZ $ CAZ=XK0*FAZ $ CYZ=XK0*FYZ                           OUT       123
      CMXZ=XK1*(MXZ+ZC*FYZ) $ CMYZ=XK1*(MYZ-ZC*FNZ) $ CMZZ=XK1*MZZ     OUT       124
      CYZ=-CYZ $ CMXZ=-CMXZ $ CMYZ=-CMYZ $ CMZZ=-CMZZ                   OUT       125
      ZZ=Z+Z0                                                           OUT       126
      WRITE (6,3360) ZZ,CNZ,CAZ,CYZ,CMXZ,CMYZ,CMZZ                     OUT       127
      GO TO 350                                                         OUT       128
  425 RETURN                                                            OUT       129
      ENTRY OUTR                                                        OUT       130
C*********************************************************************** OUT       131
C                                                                       OUT       132
C       THIS PART OF THE ROUTINE READS IN DATA USED BY OUT              OUT       133
C                                                                       OUT       134
C*********************************************************************** OUT       135
C                                                                       OUT       136
C       ZREF IS THE REFERENCE LENGTH                                    OUT       137
C       AREF IS THE REFERENCE AREA                                      OUT       138
C       ZC IS THE Z VALUE THAT MOMENT COEFFICIENTS ARE TAKEN ABOUT      OUT       139
C       Z0 IS THE OFFSET DISTANCE FROM BEGINNING OF BODY                OUT       140
C       IPCID = 0 MEANS PRINT SURFACE PRESSURE RATIO                    OUT       141
C             = 1 MEANS PRINT SURFACE PRESSURE COEFFICIENT              OUT       142
C                                                                       OUT       143
      ZREF=AREF=0. $ ZC=0. $ Z0=0. $ IPCID=0                            OUT       144
      READ (5,OUTRD)                                                    OUT       145
      RETURN                                                            OUT       146
 3200 FORMAT(1H1,36X,*A E R O D Y N A M I C   D A T A*)                 OUT       147
 3210 FORMAT(1H0,30X,*F R E E   S T R E A M   C O N D I T I O N S*)     OUT       148
 3220 FORMAT(1H ,10HMACH NO. =,1PE15.6,3X,6X,17HANGLE OF ATTACK =,      OUT       149
     1   1PE15.6,3X,19HANGLE OF SIDESLIP =,1PE15.6)                     OUT       150
 3230 FORMAT(1H ,4X,6HVINF =,1PE15.6,3X,23HTOTAL ANGLE OF ATTACK =,     OUT       151
     1   1PE15.6,3X,2X,17HAERO ROLL ANGLE =,1PE15.6)                    OUT       152
 3240 FORMAT(1H ,4X,6HPINF =,1PE15.6,3X,17X,6HDINF =,                   OUT       153
     1   1PE15.6,3X,13X,6HSINF =,1PE15.6)                               OUT       154
 3250 FORMAT(1H ,*PERFECT GAS   (GAMMA =*,1PE15.6,*)*)                  OUT       155
 3260 FORMAT(1H ,*REAL GAS   (GAS NUMBER IS*,I3,*)*)                    OUT       156
 3270 FORMAT(1H0,32X,*R E F E R E N C E   Q U A N T I T I E S*)         OUT       157
 3280 FORMAT(1H ,*REFERENCE LENGTH IS*,1PE15.6,7X,*REFERENCE AREA IS*,  OUT       158
     1   1PE15.6,7X,*Z0 IS*,1PE15.6)                                    OUT       159
 3290 FORMAT(1H0,19X,*A E R O D Y N A M I C   C O E F F I C I E N T S* OUT       160
     1   *   AT   T A R G E T E D   Z   L O C A T I O N S*)             OUT       161
```

```
      ZZ=Z+Z0                                                                  OUT       48
      IF (IPCID .EQ. 1) GO TO 100                                              OUT       49
      WRITE (6,3050) ZZ,(PB(M),M=M1,M2)                                        OUT       50
 3050 FORMAT(1H ,F9.3,15F8.3)                                                  OUT       51
      GO TO 125                                                                OUT       52
  100 DO 110 M=M1,M2                                                           OUT       53
  110 PB(M)=(PB(M)-1.)/CONT1                                                   OUT       54
      WRITE (6,3055) ZZ,(PB(M),M=M1,M2)                                        OUT       55
 3055 FORMAT(1H ,F9.3,15F8.4)                                                  OUT       56
  125 IF (M1 .GT. 1) GO TO 25                                                  OUT       57
      MCMX=MAX0(MC,MCMX)                                                       OUT       58
      WRITE(18) Z,FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ                   OUT       59
      GO TO 25                                                                 OUT       60
  200 M1=M1+15 $ IF (M1 .GT. MCMX) GO TO 250                                   OUT       61
      M1P14=M1+14 $ NPTS=0 $ REWIND 16                                         OUT       62
      GO TO 25                                                                 OUT       63
  250 XK0=2./(DINF*VINF*VINF*AREF) $ XK1=XK0/ZREF                              OUT       64
      ATTAR=ATTA*RAD $ YAWR=YAW*RAD                                            OUT       65
      ALPT=ACOS(COS(YAWR)*COS(ATTAR))/RAD                                      OUT       66
      PHIC=XINDEF $ IF (ATTA.NE.0.) PHIC=ATAN(-TAN(YAWR)/SIN(ATTAR))/RAD       OUT       67
      WRITE (6,3200)                                                           OUT       68
      WRITE (6,3210)                                                           OUT       69
      WRITE (6,3220) ACH,ATTA,YAW                                              OUT       70
      WRITE (6,3230) VINF,ALPT,PHIC                                            OUT       71
      WRITE (6,3240) PINF,DINF,SINF                                            OUT       72
      IF (NGAS .LE. 0) WRITE (6,3250) GAMMA                                    OUT       73
      IF (NGAS .GT. 0) WRITE (6,3260) NGAS                                     OUT       74
      WRITE (6,3270)                                                           OUT       75
      WRITE (6,3280) ZREF,AREF,Z0                                              OUT       76
      WRITE (6,3290)                                                           OUT       77
      WRITE (6,3300) ZC                                                        OUT       78
      WRITE (6,3320)                                                           OUT       79
      IF (NTARGET .EQ. 0) GO TO 265                                            OUT       80
      DO 260 ITARGET=1,NTARGET                                                 OUT       81
      IF (TARGETZ(ITARGET) .GE. Z) GO TO 262                                   OUT       82
  260 CONTINUE                                                                 OUT       83
      GO TO 265                                                                OUT       84
  262 NTARGET=ITARGET-1                                                        CORR1      9
  265 IF (Z .GE. ZEND) GO TO 270                                               OUT       86
      NTARGET=NTARGET+1 $ TARGETZ(NTARGET)=Z                                   OUT       87
  270 NTARGET=NTARGET+1 $ TARGETZ(NTARGET)=ZEND                                OUT       88
      I1=2 $ I2=1 $ ITARGET=1                                                  OUT       89
      REWIND 18                                                                OUT       90
      READ (18) ZS(I2),FNS(I2),FYS(I2),FAS(I2),MXS(I2),MYS(I2),MZS(I2)         OUT       91
  275 I2S=I2 $ I2=I1 $ I1=I2S                                                  OUT       92
      READ (18) ZS(I2),FNS(I2),FYS(I2),FAS(I2),MXS(I2),MYS(I2),MZS(I2)         OUT       93
      IF (EOF(18)) 325,300                                                     OUT       94
  300 IF (ITARGET .GT. NTARGET) GO TO 275                                      OUT       95
      IF (ZS(I2) .LT. TARGETZ(ITARGET)) GO TO 275                              OUT       96
      ZT=TARGETZ(ITARGET)                                                      OUT       97
      ITARGET=ITARGET+1                                                        OUT       98
      CONT1=(ZT-ZS(I1))/(ZS(I2)-ZS(I1))                                        OUT       99
      FN=FNS(I1)+(FNS(I2)-FNS(I1))*CONT1                                       OUT      100
      FY=FYS(I1)+(FYS(I2)-FYS(I1))*CONT1                                       OUT      101
      FA=FAS(I1)+(FAS(I2)-FAS(I1))*CONT1                                       OUT      102
      MX=MXS(I1)+(MXS(I2)-MXS(I1))*CONT1                                       OUT      103
      MY=MYS(I1)+(MYS(I2)-MYS(I1))*CONT1                                       OUT      104
```

```
3300 FORMAT(1H ,9X,15X,18HFORCE COEFFICIENTS,12X,                    OUT      162
     1   2X,29HMOMENT COEFFICIENTS ABOUT Z =,1PE15.6,7X,             OUT      163
     2   19HCENTERS OF PRESSURE)                                     OUT      164
3320 FORMAT(1H ,5X,4HZ-Z0,10X,2HCN,3X,10X,2HCA,3X,10X,2HCY,3X,      OUT      165
     1   9X,3HCMN,3X,9X,3HCMM,3X,9X,3HCML,3X,8X,4HXCPP,3X,8X,4HXCPY) OUT      166
3330 FORMAT(1H ,F9.3,1P8E15.5)                                       OUT      167
3340 FORMAT(1H0,28X,*Z DERIVATIVES OF FORCE AND MOMENT COEFFICIENTS*) OUT      168
3350 FORMAT(1H ,5X,*Z-Z0*,9X,*CNZ*,13X,*CAZ*,13X,*CYZ*,12X,*CMN7*,12X, OUT     169
     1   *CMMZ*,12X,*CMLZ*)                                          OUT      170
3360 FORMAT(1H ,F9.3,1P6E16.5)                                       OUT      171
     END                                                             OUT      172
```

```
      SUBROUTINE SAVE(EX,ENRUN,RAPO)                                    SAVE      2
C                                                                       SAVE      3
C      SAVE PRINTS OUT THE FIELD DATA FOR THE LAST IERRPR STEPS,        SAVE      4
C      IF THE PROGRAM BOMBS BECAUSE OF AN ERROR CONDITION.              SAVE      5
C      ALSO THE WALL PRESSURES, FORCES AND MOMENTS ARE PRINTED.         SAVE      6
C                                                                       SAVE      7
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                        NEWCOM    1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)               NEWCOM    2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)    NEWCOM    3
      COMMON CU(4,20,25),CUP(4,20,25)                                   NEWCOM    4
C   ***   END OF BLANK COMMON   ***                                     CD3CSS   32
      COMMON /CSAVE/ IERRPR,MAS                                         CSAVE     2
      COMMON /CBODY/ Z,BZZ,BPHPHI,BZPHI,TANCO,DELZ                      CBODY     2
     1   ,PHI(25),B(25),BZ(25),BPHI(25),COSPHI(25),SINPHI(25)          CBODY     3
C                                                                       SAVE      9
      DIMENSION EX(17)                                                  SAVE     10
      ENDFILE 16                                                        SAVE     11
      IF (IERRPR .LE. 0) GO TO 75                                       SAVE     12
      DO 25 I=1,IERRPR                                                  SAVE     13
   25 BACKSPACE 16                                                      SAVE     14
      BACKSPACE 16                                                      SAVE     15
      DO 50 I=1,IERRPR                                                  SAVE     16
      READ (16) NC,MC,ATTA,YAW,ACH,GAMMA,PINF,DINF,PHIO,K,Z             SAVE     17
     A  ,NGAS,NTEST,RRX                                                 SAVE     18
     1  ,FN,FY,FA,MX,MY,MZ,FNZ,FYZ,FAZ,MXZ,MYZ,MZZ                     SAVE     19
     2   ,(PHI(M),C(M),CZ(M),CPHI(M),M=1,MC)                           SAVE     20
     3   ,((R(N,M),U(N,M),V(N,M),W(N,M),P(N,M),D(N,M),M=1,MC),N=1,NC)  SAVE     21
      DO 40 M=1,MAS                                                     SAVE     22
   40 CALL BODY(M)                                                      SAVE     23
   50 CALL FIELD                                                        SAVE     24
   75 CALL OUT                                                          SAVE     25
      STOP                                                              SAVE     26
      END                                                               SAVE     27
```

```
      SUBROUTINE TRANFD                                                 TRANFD     2
                                                                        TRANFD     3
C                                                                       TRANFD     4
C      TRANFD DEFINES QUANTITIES NEEDED BY SUBROUTINE                   TRANFD     5
C      TRANF WHEN THE USER READS IN THE SF(X,Y,Z) DATA POINTS           TRANFD     6
C                                                                       NEWCOM     8
      COMMON /CTRANF/ NSFD,SFD(20),SFXD(20),SFXXD(20)                   TRANFD     9
C                                                                       TRANFD    10
      READ (5,2000) (SFD(N),N=1,NSFD)                                   TRANFD    11
 2000 FORMAT(5F10.0)                                                    TRANFD    12
      NSFDM1=NSFD-1                                                     TRANFD    13
      DX=1./FLOAT(NSFDM1)                                               TRANFD    14
      TWODX=1./(2.*DX) $ DXSQ=1./DX**2                                  TRANFD    15
      DO 50 N=2,NSFDM1                                                  TRANFD    16
      SFXD(N)=(SFD(N+1)-SFD(N-1))*TWODX                                 TRANFD    17
   50 SFXXD(N)=(SFD(N+1)-2.*SFD(N)+SFD(N-1))*DXSQ                       TRANFD    18
C                                                                       TRANFD    19
C      SFXXD IS ASSUMED LINEAR ON (0,2DX) AND (1-2DX,1)                 TRANFD    20
C                                                                       TRANFD    21
      SFXXD(1)=2.*SFXXD(2)-SFXXD(3)                                     TRANFD    22
      SFXXD(NSFD)=2.*SFXXD(NSFDM1)-SFXXD(NSFD-2)                        TRANFD    23
      SFXD(1)=SFXD(2)-.5*DX*(SFXXD(1)+SFXXD(2))                         TRANFD    24
      SFXD(NSFD)=SFXD(NSFDM1)+.5*DX*(SFXXD(NSFD)+SFXXD(NSFDM1))         TRANFD    25
      WRITE (6,3300)                                                    TRANFD    26
 3300 FORMAT(1H1,4X,*N*,15X,*SF*,17X,*SFX*,16X,*SFXX*)                  TRANFD    27
      DO 125 N=1,NSFD                                                   TRANFD    28
  125 WRITE (6,3400) N,SFD(N),SFXD(N),SFXXD(N)                          TRANFD    29
 3400 FORMAT(1H ,I5,1P4E20.6)                                           TRANFD    30
      RETURN                                                            TRANFD    31
      END
```

```
      SUBROUTINE TRANGD                                                  TRANGD     2
C                                                                        TRANGD     3
C      TRANGD DEFINES QUANTITIES NEEDED BY SUBROUTINE                    TRANGD     4
C      TRANG WHEN THE USER READS IN THE PHI VALUES                       TRANGD     5
C                                                                        TRANGD     6
      COMMON NC,MC,K,PINF,DINF,PHIO,IDYAW,PI,RAD                         NEWCOM     1
      COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(20,25)                NEWCOM     2
      COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)     NEWCOM     3
      COMMON CU(4,20,25),CUP(4,20,25)                                    NEWCOM     4
C   ***    END OF BLANK COMMON    ***                                    CD3CSS    32
      COMMON /CTRANG/ NSGD,SGD(25),SGYD(25),SGYYD(25)                    NEWCOM     6
     1   ,GYMDY,GYYMDY,GY1PDY,GYY1PDY                                    CTRANG     3
     2   ,MCP                                                            NEWCOM     7
C                                                                        TRANGD     9
      READ (5,2000) (SGD(M),M=1,NSGD)                                    TRANGD    10
 2000 FORMAT(5F10.0)                                                     TRANGD    11
      NSGDM1=NSGD-1                                                      TRANGD    12
      DY=1./FLOAT(NSGDM1)                                                TRANGD    13
      TWODY=1./(2.*DY) $ DYSQ=1./DY**2                                   TRANGD    14
      DO 25 M=1,NSGD                                                     TRANGD    15
   25 SGD(M)=SGD(M)/SGD(NSGD)                                            TRANGD    16
      DO 50 M=2,NSGDM1                                                   TRANGD    17
      SGYD(M)=(SGD(M+1)-SGD(M-1))*TWODY                                  TRANGD    18
   50 SGYYD(M)=(SGD(M+1)-2.*SGD(M)+SGD(M-1))*DYSQ                        TRANGD    19
      IF (PHIO .LE. 2.*PI-1.E-6) GO TO 75                                TRANGD    20
C                                                                        TRANGD    21
C      NOTE THAT SGD(Y+1)=SGD(Y)+1  FOR NON-SYMMETRIC PROBLEM (PHIO=360) TRANGD    22
C                                                                        TRANGD    23
      SGYD(1)=SGYD(NSGD)=(SGD(2)-SGD(NSGD-1)+1.)*TWODY                   TRANGD    24
      SGYYD(1)=SGYYD(NSGD)=(SGD(2)-2.*SGD(1)+SGD(NSGD-1)-1.)*DYSQ        TRANGD    25
      GO TO 100                                                          TRANGD    26
C                                                                        TRANGD    27
C      NOTE THAT FOR SYMMETRIC PROBLEM (PHIO=180)                        TRANGD    28
C      SGYYD IS ASSUMED LINEAR ON (-DY,2DY) AND (1-2DY,1+DY)             TRANGD    29
C                                                                        TRANGD    30
   75 SGYYD(1)=2.*SGYYD(2)-SGYYD(3)                                      TRANGD    31
      SGYYD(NSGD)=2.*SGYYD(NSGDM1)-SGYYD(NSGD-2)                         TRANGD    32
      GYYMDY=2.*SGYYD(1)-SGYYD(2)                                        TRANGD    33
      GYY1PDY=2.*SGYYD(NSGD)-SGYYD(NSGDM1)                               TRANGD    34
      SGYD(1)=SGYD(2)-.5*DY*(SGYYD(1)+SGYYD(2))                          TRANGD    35
      SGYD(NSGD)=SGYD(NSGDM1)+.5*DY*(SGYYD(NSGD)+SGYYD(NSGDM1))          TRANGD    36
      GYMDY=SGYD(2)-2.*DY*SGYYD(1)                                       TRANGD    37
      GY1PDY=SGYD(NSGDM1)+2.*DY*SGYYD(NSGD)                              TRANGD    38
  100 WRITE (6,3300)                                                     TRANGD    39
 3300 FORMAT(1H1,4X,*M*,15X,*PHI*,18X,*SG*,17X,*SGY*,16X,*SGYY*)         TRANGD    40
      M=0 $ WRITE (6,3350) M,GYMDY,GYYMDY                                TRANGD    41
 3350 FORMAT(1H ,I5,40X,1P3E20.6)                                        TRANGD    42
      PHIOD=PHIO/RAD                                                     TRANGD    43
      DO 125 M=1,NSGD                                                    TRANGD    44
      PHIM=SGD(M)*PHIOD                                                  TRANGD    45
  125 WRITE (6,3400) M,PHIM,SGD(M),SGYD(M),SGYYD(M)                      TRANGD    46
 3400 FORMAT(1H ,I5,1P4E20.6)                                            TRANGD    47
      M=NSGD+1 $ WRITE (6,3350) M,GY1PDY,GYY1PDY                         TRANGD    48
      RETURN                                                             TRANGD    49
      END                                                                TRANGD    50
```

```
      SUBROUTINE DMPSQRT(NAME,KNT,Z,K,M,N,VALUE)                        DMPSQRT    2
C                                                                       DMPSQRT    3
C      NAME IS THE NAME OF THE ROUTINE                                  DMPSQRT    4
C      KNT IS THE NUMBER FROM WHICH NAME WAS CALLED                     DMPSQRT    5
C      Z IS THE Z VALUE                                                 DMPSQRT    6
C      K IS THE STATION NO.                                             DMPSQRT    7
C      M IS THE PLANE NO.                                               DMPSQRT    8
C      N IS THE RADIAL POINT NO.                                        DMPSQRT    9
C      VALUE IS THE ARGUMENT OF SQRT ROOT                               DMPSQRT   10
C                                                                       DMPSQRT   11
      WRITE (6,3000) NAME,KNT,VALUE,Z,K,M,N                             DMPSQRT   12
 3000 FORMAT(1H1,*IN ROUTINE *,A10,* AT CALL NO.*,I3,2X,                DMPSQRT   13
     1   *NEGATIVE SQRT ROOT OF*,1PE15.6,                               DMPSQRT   14
     2   * FOR Z,K,M,N*,1PE15.6,3I5)                                    DMPSQRT   15
      CALL SAVE(DUM,DUM,DUM)                                            DMPSQRT   16
      STOP                                                              DMPSQRT   17
      END                                                               DMPSQRT   18
```

DISTRIBUTION

	Copies
Commander Naval Sea Systems Command Headquarters Department of the Navy Washington, D.C. 20360 Chief Technical Analyst SEA 05121 SEA 033 SEA 031 SEA 09G32 SEA 035	2
Commander Naval Air Systems Command Headquarters Department of the Navy Washington, D.C. 20360 AIR 03B AIR 03C AIR 320 AIR 320C AIR 310 AIR 50174	2
Office of Naval Research 800 N. Quincy Street Arlington, Virginia 22217 ONR 100 M. Cooper, 430B Dr. Leila Bram, 432	2
Commander David Taylor Naval Ship Research and Development Center Bethesda, Maryland 20034 Central Library (5641) Aerodynamics Lab. (5643)	
Commander Naval Weapons Center China Lake, California 93555 Technical Library (533) Code 406 R. E. Meeker (4063) W. Thielbahr	

DISTRIBUTION

Copies

Director
U.S. Naval Research Laboratory
Washington, D.C. 20390
  Library
  Code 6503

ICASE, NASA Langley
MS 132C
Hampton, Virginia 23665
  Dr. James Ortega

NASA
Langley Research Center
Langley Station
Hampton, Virginia 23665
  MS/185 Technical Library
  Aero & Space Mech Div.
  Dennis Bushnell
  Ivan Beckwith
  R. Trimpi
  Julius Harris

NASA
Lewis Research Center
21000 Brookpart Road
Cleveland, Ohio 44135
  Library 60-3
  Chief, Wind Tunnel & Flight Div.

NASA
George C. Marshall Space Flight Center
Huntsville, Alabama 35812
  Mr. T. Reed, R-AERO-AU
  Mr. W. K. Dahm, BB31

NASA
600 Independence Avenue, S. W.
Washington, D. C. 20546
  F. C. Schwenk, Director,
    Research (Code RR)

NASA
P.O. Box 33
College Park, Maryland 20740

DISTRIBUTION

Copies

Director
Defense Research and Engineering (DDR&E)
Room 3E-1063, The Pentagon, Stop 103
Washington, D.C. 20301
  Technical Library

Defense Documentation Center
Cameron Station
Alexandria, Virginia 22314 — 12

Commander (5632.2)
Naval Missile Center
Point Mugu, California 93041
  Technical Library

Commanding Officer
USA Aberdeen Research and
  Development Center
Aberdeen Proving Ground
Maryland 21005
  STEAP-TL
  AMXRD-XSE

Director
Strategic Systems Project Office
Department of the Navy
Washington, D.C. 20390
  NSP-2722

Director of Intelligence
Headquarters, USAF (AFNINDE)
Washington, D.C. 20330
  AFOIN-3B

Los Angeles Air Force Station
SAMSO/DYAE
P.O. Box 92960
Worldway Postal Center
Los Angeles, California 90009
  Code RSSE
  Code RSSM
  LT D. Hall, RSSM

DISTRIBUTION

Copies

Headquarters, Arnold Engineering
Development Center
Arnold Air Force Station
Tennessee 37389
Library/Documents
R. W. Henzel, TD
CAPT C. Tirres/DYR
CAPT K. Kushman

von Karman Gas Dynamics Facility
ARO, Inc.
Arnold Air Force Station
Tennessee 37389
Dr. J. D. Whitfield
Dr. J. Adams

Commanding Officer
Harry Diamond Laboratories
Washington, D.C. 20438
Library

Commanding General
U.S. Army Missile Command
Redstone Arsenal
Alabama 35809
AMSMI-RR
Chief, Document Section
AMSMI-RDK, R. A. Deep
AMSMI-RDK, T. R. Street

Department of the Army
Office of the Chief of
Research and Development
ABMDA, The Pentagon
Washington, D.C. 20350

Commanding Officer
Picatinny Arsenal
Dover, New Jersey 07801
SMUPA-VC-3 (A. A. Loeb)

Commander (ADL)
Naval Air Development Center
Johnsville, Pennsylvania 18974
Dr. R. K. Lobb

DISTRIBUTION

Copies

Air Force Weapons Laboratory
Kirtland Air Force Base
Albuquerque, New Mexico 87117
  CAPT Tolman/SAS
  Tech. Library (SUL)

U.S. Army Ballistic Missile
  Defense Agency
1300 Wilson Boulevard
Arlington, Virginia 22209

The Johns Hopkins University
  (C/NOw 7386)
Applied Physics Laboratory
Johns Hopkins Road
Laurel, Maryland 20810
  Document Library
  Dr. F. Hill
  Dr. L. L. Cronvich
  Dr. J. C. W. Rogers

Director, Defense Nuclear Agency
Headquarters, DASA
Washington, D.C. 20305
  STSP (SPAS)

Commanding Officer
Naval Intelligence Support Center
4301 Suitland Road
Washington, D.C. 20390

Department of Aeronautics
DFAN
USAF, Academy
Colorado 80840
  COL D. H. Daley
  CAPT J. Williams
  Library

Armament Development and Test Center
Eglin AFB, Florida 32542
  Technical Library, DLOSL

Commander
U.S. Army Natick Development Center
AMSNM-UBS
Natick, Massachusetts 01760
  AMXNM-UBS
  G. A. Barnard

DISTRIBUTION

Copies

NASA Ames Research Center
Moffett Field, California 94035
Dr. M. Horstman
P. Kutler
J. Rakich
R. MacCormack
Library

AFFDL
Wright Patterson Air Force Base
Ohio 45433
Mr. Melvin Buck
D. J. Harney
W. Hankey
J. Shang
K. Stetson
Art Lewis

Wright Aeronautical Laboratories
Wright-Patterson Air Force Base
Dayton, Ohio 45433
Technical Library

Air Force Materials Laboratory
Wright Patterson Air Force Base
Dayton, Ohio 45433
Dr. William Kessler

Naval Research Laboratory
Washington, D.C.
Dr. J. Boris

ONR Branch Office/Pasadena
1030 East Green Street
Pasadena, California 91101
Dr. Richard Lau

Aerospace Engineering Program
University of Alabama
P.O. Box 6307
University of Alabama 35486
Prof. W. K. Rey, Chm.

AME Department
University of Arizona
Tucson, Arizona 85721
Dr. L. B. Scott

DISTRIBUTION

Copies

Polytechnic Institute of New York
Graduate Center Library
Route 110, Farmingdale
Long Island, New York 11735
Dr. R. Cresci
Prof. Gino Moretti

Polytechnic Institute of New York
Spicer Library
333 Jay Street
Brooklyn, New York 11201
Reference Dept.

California Institute of Technology
Graduate Aeronautical Laboratories Aero.
Pasadena, California 91109
Librarian
Dr. A. Roshko
Prof. H. B. Keller,
Dept. of Mathematics

University of California
Dept. of Mechanical Engineering
Berkeley, CA 94720
Prof. R. Greif

GASDYNAMICS
University of California
Richmond Field Station
1301 South 46th Street
Richmond, California 94804
A. K. Oppenheim

Dept. of Aerospace Engineering
University of Southern California
University Park
Los Angeles, California 90007
Dr. John Laufer

University of California, San Diego
Department of Aerospace and
Mechanical Engineering Sciences
LaJolla, California 92037
Dr. P. A. Libby

DISTRIBUTION

Copies

Case Western Reserve University
Division of Fluid, Thermal and
Aerospace Engineering
Cleveland, Ohio 44106
Dr. Eli Reshotko

The Catholic University
Washington, D.C. 20017
Dr. C. C. Chang

University of Cincinnati
Cincinnati, Ohio 45221
Department of Aerospace Engineering
Dr. Arnold Polak
Dr. Michael Werle

Department of Aerospace Engineering
Sciences
University of Colorado
Boulder, Colorado 80302

Cornell University
Graduate School of Aero. Engineering
Ithaca, New York 14850
Prof. A. R. George

University of Delaware
Mechanical and Aeronautical
Engineering Dept.
Newark, Delaware 19711
Dr. James E. Danberg

Georgia Institute of Technology
225 North Avenue, N. W.
Atlanta, Georgia 30332
Dr. Arnold L. Ducoffe

Technical Reports Collection
Gordon McKay Library
Harvard University
Div. of Engineering and Applied Physics
Pierce Hall, Oxford Street
Cambridge, Massachusetts 02138
Dr. George Carrier

DISTRIBUTION

Copies

Illinois Institute of Technology
3300 South Federal
Chicago, Illinois 60616
Dr. M. V. Morkovin

University of Illinois
101 Transportation Bldg.
Urbana, Illinois 61801
Aeronautical and Astronautical
Engineering Dept.

Iowa State University
Ames, Iowa 50010
Aerospace Engineering Dept.

The Johns Hopkins University
Baltimore, Maryland 21218
Prof. S. Corrsin

University of Kentucky
Wenner-Gren Aero. Lab.
Lexington, Kentucky 40506
C. F. Knapp

Department of Aero. Engineering, ME 106
Louisiana State University
Baton Rouge, Louisiana 70803
Dr. P. H. Miller

University of Maryland
College Park, Maryland 20740
Dr. S. I. Pai, Institute for
Fluid Dynamics and
Applied Mathematics
Dr. John D. Anderson, Jr.
Dept. of Aerospace Engineering

Michigan State University
East Lansing, Michigan 48823
Library Documents Dept.

Massachusetts Institute of Technology
Cambridge, Massachusetts 02139
Prof. A. H. Shapiro, Head
Mech. Engr. Dept.
Aero. Engineering Library
Prof. R. F. Probestein
Dr. E. E. Covert
Aerophysics Laboratory

DISTRIBUTION

Copies

University of Michigan
Ann Arbor, Michigan 48104
Dr. M. Sichel, Dept. of Aero. Engr.
Engineering Library
Aerospace Engineering Library
Mr. C. Cousineau, Engin-Trans Lib.

Serials and Documents Section
General Library
University of Michigan
Ann Arbor, Michigan 48104

Mississippi State Univ.
Dept. of Aerophysics and
Aerospace Engineering
P.O. Drawer A
State College, Mississippi 39762
Mr. Charles B. Cliett

U.S. Naval Academy
Annapolis, Maryland 21402
Engineering Dept., Aerospace Div.

Library, Code 2124
U.S. Naval Postgraduate School
Monterey, California 93940
Technical Reports Section
Prof. Carroll Wilde, Ch. Mathematics Dept.

North Carolina State College
Raleigh, North Carolina 27607
Dr. F. R. DeJarnette

D. H. Hill Library
North Carolina State Univ.
P.O. Box 5007
Raleigh, North Carolina 27607

University of North Carolina
Cahpel Hill
North Carolina 27514
Dept. of Aero. Engineering

Northwestern University
Technological Institute
Evanston, Illinois 60201
Dept. of Mech. Engineering
Library

DISTRIBUTION

Copies

Dept. of Aero-Astro Engineering
Ohio State University
2036 Neil Avenue
Columbus, Ohio 43210
  Engineering Library

The Pennsylvania State Univ.
University Park, Pennsylvania 18602
  Dept. of Aero. Engr.
    Hammond Bldg.
  Library, Documents Section

Bevier Engr. Library
126 Benedum Hall
University of Pittsburgh
Pittsburgh, Pennsylvania 15261

Princeton University
Aerospace & Mech. Science Dept.
D-214 Engr. Quadrangel
Princeton, New Jersey 08540
  Dr. I. E. Vas
  Dr. W. D. Hayes

Purdue University
School of Aeronautical and
  Engineering Sciences
Lafayette, Indiana 47907
  Dr. B. Reese, Head, Dept. of
    Aero. and Astro.
  Dr. C. P. Kentzer

Rensselaer Polytechnic Institute
Troy, New York 12181
  Dept. of Aeronautical
    Engineering and Astronautics

Department of Mechanical
  Industrial and Aerospace Engr.
Rutgers - The State University
New Brunswick, New Jersey 08903
  Dr. R. H. Page

Stanford University
Stanford, CA 94305
  Librarian, Dept. of Aeronautics
    and Astronautics
  Prof. J. Oliger, Dept. of Computer Sciences

DISTRIBUTION

Copies

Stevens Institute of Technology
Hoboken, New Jersey 07030
Mechanical Engineering Dept.
Library

The University of Texas at Austin
Applied Research Laboratories
P.O. Box 8029
Austin, Texas 78712
Director
Engr. S.B.114B/Dr. Friedrich

University of Toledo
2801 W. Bancroft
Toledo, Ohio 43606
Dept. of Aero Engineering

University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia 22091
Dr. I. D. Jacobson

University of Washington
Seattle, Washington 98105
Engineering Library
Dept. of Aeronautics and Astronautics

West Virginia University
Morgantown
West Virginia 26506
Library

Federal Reports Center
University of Wisconsin
Mechanical Engineering Building
Madison, Wisconsin 53706
S. Reilly

Los Alamos Scientific Laboratory
P.O. Box 1663
Los Alamos, New Mexico 87544
Reports Library

Institute for Defense Analyses
400 Army-Navy Drive
Arlington, Virginia 22202
Classified Library

DISTRIBUTION

Copies

Kaman Sciences Corporation
P.O. Box 7463
Colorado Springs, Colorado 80933
Library
Mr. F. Barbera
Mr. D. Foxwell

Kaman Sciences Corporation
Avidyne Division
83 Second Avenue
Burlington, Massachusetts 01803
Dr. J. R. Ruetenik

Rockwell International
B-1 Division
Technical Information Center
(BA08) International Airport
Los Angeles, California 90009

Rockwell International Corporation
Technical Information Center
4300 E. Fifth Avenue
Columbus, Ohio 43216

M.I.T. Lincoln Laboratory
P.O. Box 73
Lexington, Massachusetts 02173
Library A-082
Dr. A. B. Wardlaw 5

The RAND Corporation
1700 Main Street
Santa Monica, California 90406
Library - D

Aerojet Electrosystems Co.
1100 W. Hollyvale Avenue
Azusa, California 91702
Engineering Library

The Boeing Company
P.O. Box 3999
Seattle, Washington 98124
87-67

United Aircraft
Research Laboratories
East Hartford
Connecticut 06108
Dr. William M. Foley

DISTRIBUTION

Copies

United Aircraft Corporation
400 Main Street
East Hartford, Connecticut 06108
Library

Hughes Aircraft Company
Centinela at Teale
Culver City, California 90230
Company Tech. Doc. Center
6/E11, B.W. Campbell

Lockheed Missiles and Space Co.
P.O. Box 504
Sunnyvale, California 94086
G. T. Chrusciel
R. Nelson
P. D. Thomas
Charles Lee

Lockheed Missiles and Space Co.
3251 Hanover Street
Palo Alto, California 94304
Technical Information Center

Lockheed-California Co.
Burbank, California 91503
Central Library Dept.

Vice President and Chief Scientist
Dept. 03-10
Lockheed Aircraft Corp.
P.O. Box 551
Burbank, California 91503

Martin Marietta Corporation
P.O. Box 988
Baltimore, Maryland 21203
Science-Technology Library
(Mail No. 398)

Martin Company
3211 Trade Winds Trail
Orlando, Florida 32805
Mr. H. J. Diebolt

General Dynamics
P.O. Box 748
Fort Worth, Texas 76101
Research Library 2246
George Kaler, Mail Zone 2880

DISTRIBUTION

Copies

Calspan Corporation
4455 Genesee Street
Buffalo, New York 14221
Library

Air Force Univ. Library
(SE) 63-578
Maxwell Air Force Base
Alabama 36112

McDonnell Company
P.O. Box 516
St. Louis, Missouri 63166
R. D. Detrich, Dept. 209
Bldg. 33

McDonnell Douglas Astronautics
Company - West
5301 Bolsa Avenue
Huntington Beach, California 92647
368, B4A0
J. S. Murphy, A3
J. Copper
J. Xerikos
W. Shaw

Fairchild Hiller
Republic Aviation Division
Farmingdale, New York 11735
Engineering Library

General Applied Science Labs. Inc.
Merrick and Stewart Avenues
Westbury, Long Island
New York 11590
Dr. F. Lane
L. M. Nucci

General Electric Company
R&D Lab. (Comb. Bldg.)
Schenectady, New York 12301
Dr. H. T. Nagamatsu

DISTRIBUTION

Copies

The Whitney Library
General Electric Research and
Development Center
The Knolls, K-1
P.O. Box 8
Schenectady, New York 12301
M. F. Orr, Manager

General Electric Company
Missile and Space Division
P.O. Box 8555
Philadelphia, Pennsylvania 19101
MSD Library
Dr. J. D. Stewart

General Electric Company
AEG Technical Information
Center, N-32
Cincinnati, Ohio 45215

General Electric Company
Reentry & Environmental Systems
Division
3198 Chestnut Street
Philadelphia, Pennsylvania 19101
W. Daskin
R. Neff
C. Kyriss
A. Martellucci

AVCO-Everett Research Lab.
2385 Revere Beach Parkway
Everett, Massachusetts 02149
Library
Dr. George Sutton

LTV Aerospace Corporation
Vought Aeronautics Division
P.O. Box 5907
Dallas, Texas 75222
Unit 2-51131 (Library)

LTV Aerospace Corporation
Missiles and Space Division
P.O. Box 6267
Dallas, Texas 75222
MSD-T-Library

DISTRIBUTION

Copies

Northrop Norair
3901 West Broadway
Hawthorne, California 90250
Tech. Info. 3360-32

Government Documents
The Foundren Library
Rice Institute
P.O. Box 1892
Houston, Texas 77001

Grumman Aircraft Engineering Corporation
Bethpage, Long Island
New York 11714
Dr. R. E. Melnik

Marquardt Aircraft Corp.
16555 Saticoy Street
Van Nuys, California 91409
Library

ARDE Associates
P.O. Box 286
580 Winters Avenue
Paramus, New Jersey 07652
Librarian

Aerophysics Company
3500 Connecticut Ave., N. W.
Washington, D.C. 20003
Mr. G. D. Boehler

Aeronautical Research Associates
of Princeton
50 Washington Road
Princeton, New Jersey 08540
Dr. C. duP. Donaldson

General Research Corporation
5383 Hollister Avenue
P.O. Box 3587
Santa Barbara, California 93105
Technical Information Office

DISTRIBUTION

Copies

Sandia Laboratories
Box 5800
Albuquerque, New Mexico 87115
Dr. C. Peterson
Dr. G. W. Stone
Dr. K. Touryan
Dr. R. Eaton
Kurt Putz, Div. 1333

Hercules Incorporated
Allegany Ballistics Lab.
P.O. Box 210
Cumberland, Maryland 21502
Library

General Electric Company
P.O. Box 2500
Daytona Beach, Florida 32015
Dave Hovis, Room 4109

TRW Systems Group
1 Space Park
Redondo Beach, California 90278
Tech. Library/Doc. Acquisitions

Stanford Research Institute
333 Ravenswood Avenue
Menlo Park, California 94025
Dr. G. Abrahamson

Hughes Aircraft Company
P.O. Box 3310
Fullerton, California 92634
Tech. Library, 600-C222

Westinghouse Electric Corp.
Astronuclear Laboratory
P.O. Box 10864
Pittsburgh, Pennsylvania 15236
Library

University of Tennessee
Space Institute
Tullahoma, Tennessee 37388
Prof. J. M. Wu

CONVAIR Division of General Dynamics
Library and Information
P.O. Box 12009
San Diego, California 92112

DISTRIBUTION

Copies

CONVAIR Division of General Dynamics
P.O. Box 80986
San Diego, California 92138
Dr. J. Raat, Zone 640-02
Research Library

AVCO Missiles Systems Division
201 Lowell Street
Wilmington, Massachusetts 01887
E. E. H. Schurmann
N. Tyson
H. Rosenbaum

Chrysler Corporation
Space Division
P.O. Box 29200
New Orleans, LA 70129
N. D. Kemp 2910
E. A. Rawls, Dept. 2920

General Dynamics
Pomona Division
P.O. Box 2507
Pomona, California 91766
Tech. Doc. Center (6-20)

Philco-Ford Corporation
Aeroneutronic Division
Newport Beach, California 92660
Dr. A. Demetriades

Raytheon Company
Missile Systems Division
Hartwell Road
Bedford, Maine 01730
D. P. Forsmo

TRW Systems Group
Space Park Drive
Houston, Texas 77058
M. W. Sweeney, Jr.

Marine Bioscience Laboratory
513 Sydnor Street
Ridgecrest, California 93555
Dr. A. C. Charters

DISTRIBUTION

Copies

University of California -
Los Angeles
Dept. of Mechanics & Structures
Los Angeles, California 90024
  Prof. J. D. Cole

University of Wyoming
University Station
P.O. Box 3295
Laramie, Wyoming 82070
  Head, Dept. Mech. Eng.

Applied Mechanics Review
Southwest Research Institute
8500 Culebra Road
San Antonio, Texas 78228

American Institute of Aeronautics
  and Astronautics
New York, New York 10019
  J. Newbauer

Technical Information Services
AIAA
750 Third Avenue
New York, New York 10017
  Miss P. Marshall

Faculty of Aeronautical Systems
Univ. of West Florida
Pensacola, Florida 32504
  Dr. R. Fledderman

Space Research Corporation
Chittenden Bank Building
North Troy, Vermont 05859
  Library
  J. A. Finkel

The Aerospace Corporation
P.O. Box 92957
Los Angeles, CA 90009
  E. Ndefo
  J. M. Lyons

DISTRIBUTION

Copies

Notre Dame University
Notre Dame, Indiana 46556
Dept. of Aero. Engineering
College of Engineering
Library

Acurex Corp. Aerotherm
485 Clyde Avenue
Mt. View, California 94042
M. Abbett

Prototype Development Assoc.
1740 Garry Avenue
Suite 201
Santa Ana, California 92705
Mr. J. Dunn

Courant Institute
New York University
251 Mercer Street
New York, New York 10012
Prof. Peter Lax
Prof. Heinz O. Kreiss
Prof. E. Isaacson
Dr. Eli Turkel
Dr. Amiram Harten

Mathematics Research Center
U.S. Army
University of Wisconsin
Madison, Wisconsin 53706
Prof. Seymour V. Parter

TRW
One Space Park
Redondo Beach, California 92078
B. Pearce
Aerodynamics Dept.
Garth W. Lippmann, Bldg. R-5, Rm. 2230

Lockheed Missiles & Space Co.
Continental Bldg., Suite 445
El Segundo, California 90245
R. Fortune

DISTRIBUTION

Copies

Department of Mathematics
University of California at Berkeley
Berkeley, California 94720
Prof. A. Chorin

Science Applications Inc.
101 Continental Bldg.
El Segundo Blvd.
El Segundo, California 90245
L. Cassel

University of Tennessee
Department of Mathematics
Knoxville, Tennessee 37916
Prof. Max Gunzburger